
ATARI
ST
COMPUTER
April ´88
DM 7,-
4
Ös. 56,- Sfr. 7,-
HÄTTE GUTENBERG EINEN ST GEHABT ...
-Einbinden eigener GEM-Zeichensätze
NEUER VT52-EMULATOR
-Schnellere Bildschirmausgabe und viele weitere Features
DRUCKER-SCANNER IM SELBSTBAU
ORDNUNG AUF DER FESTPLATTE
mit HD-Pack
NEUE SERIE
Programmierung relationaler Datenbanken

Das klassische Dilemma

Liebe Leser!

Aurea prima sata est. Sehr frei übersetzt: Früher ging's uns noch gold. Jeder mochte jeden, keiner tat einem anderen ein Leid. Doch eines Tages begab es sich, daß ein paar Menschen zusammenfanden, die sich unsterblich verliebt hatten - in einen Computer. Sie gründeten frohgemut eine Zeitschrift für diese neue machina arithmetica und nannten sich fürderhin Redaktion. Und die Pinien stiegen herab von den Hügeln und Bergen und ließen sich zu Papier verarbeiten und mit Druckerschwärze veredeln, um geduldig das in die Welt hinauszutragen, was unsere Redakteure in selbige zu setzen trachteten.

Man bemühte sich stets redlich, es jedem recht zu machen. Neue Soft- und Hardware wollte wohlwollend gewogen werden, zu dem Ende, daß sich die Leserschaft, wie sich die Menschen draußen im Lande jetzt nennen durften, davon ein Bild machen könne. Worin die Kandidaten fehlten, sollte nicht beschönigt oder sogar verschwiegen werden, aber gleichzeitig sollte auch keine harsche Kritik da geübt werden, wo es nur an Kleinigkeiten hakte. Man wußte, wieviel Fleiß und Geduld Softwerker auf ihre Programme und Hardwerker auf Steckkarten und Peripheriegeräte verwenden mußten. Und man wußte, daß diese jetzt auf eine gute Presse bauten.

Gewiß war man sich auch bewußt, daß ein loyaler Hofbericht einen königlichen Werbeauftrag bescheren konnte, der mit einem Beutel puren Goldes dotiert wäre, während ein Verriß allenthalben tiefen Gram säte, so daß man es beim König Werbekunden selten noch über eine Anstellung als Hofnarr hinausbrächte.

Diese moralisch-monetäre Zwickmühle zerrte an den Drahtseilen unserer Redakteure, aber sie nahmen sich vor, die beschwerliche Route über den Grat zu nehmen, gleichwohl dessen gewahr, daß es sich im Tal der Könige bequemer liefe. Ruhe und Geflissenheit setzten sie sich zum Ziel, alldieweil die stillsten Wasser stets am tiefsten gründen.

Schon ging es ihnen nicht mehr gold! Kam ihnen ein Prüfling unter ihre unbestechlichen Augen, der nach Entfernen blendender Verpackung grobe Mängel offenbaren mußte, und entdeckten sie eingedenk ihres Wahlspruchs seine Larve dem Publikum, so kam es oft, daß sich in der Folge die Gestirne verfinsterten und sich die Erde zu Füßen unserer Redakteure zumindest einen kleinen, kaum sichtbaren Spalt auftat, bereit, sie alle zu verschlingen. Teléphonos, der Gott der Ferngespräche, zürnte. "Das ist doch alles nicht wahr", "Ihr Tester kommt wohl vom Sonntagsblatt", "Der hat keine Ahnung wovon er schreibt" oder anders schrien betroffene Werbekundenkönige wutschäumend und ohrenbetäubend in ihre magischen Muscheln - so laut, daß selbst der benannte Gott und der Postminister aufschraken.

Und dann, eines Morgens, geschah es, daß eine Botin mit einem auf rätselhafte Weise hellenisch anmutendem Gesichtsausdruck in nicht ganz zeitgemäßer Gewandung seltsam feierlich in die Mitte einer in schauriger Andacht erstarrten Redaktion schritt und eine alte, verstaubte Büchse auf den Boden stellte, um schon im nächsten Augenblick wieder vor aller Augen zu entschwinden. Naja, die Tür war offen.

Als der Chefredakteur endlich mittels moderner Alchemie den Deckel gesäubert hatte, traten ein paar griechische Zeichen zum Vorschein, die ein Philologe in der entsetzten Runde endlich zu entschlüsseln vermochte. Was da in zierlicher Schrift stand, war: "Gruß und Kuß, Pandora!"

Indes, seien Sie versichert, liebe Leser, daß Ihnen die ST-Redaktion weiter sachlich als Ratgeber zur Seite stehen wird. Jedenfalls solange, bis unser Philologe, der seit jenen mysteriösen Geschehnissen völlig verwirrt ist, diesen verflixten Büchsenöffner herausrückt!

Ihr

Meinhard Ullrich

I N



SOFTWARE

HARDWARE

GRUNDLAGEN

Schönere Programme durch GEM-Ze ichensätze nach Wahl

GEM bietet einige sehr interessante Möglichkeiten, die auf dem ST leider noch gar nicht richtig ausgenutzt werden. Ein wichtiger Punkt dabei sind die Textdarstellungen. Da jeder Text im Grafikmodus ausgegeben wird, bietet GEM (VDI) die Möglichkeit, verschiedene Textattribute (fett usw.) einzustellen und die Textgröße zu wählen. Davon machen ja auch schon viele Programme (v.a. im Grafikbereich) reichlich Gebrauch. Die Entwickler von GEM sind aber noch einen sehr großen Schritt weitergegangen. Man kann außer den Systemzeichensätzen noch andere Fonts von Diskette laden (sollte man wenigstens können). Dabei tritt nur ein großes Problem auf: die Implementation auf dem ATARI ST.

Seite 14

Extended VT 52-Emulator

Daß man den eingebauten VT 52-Emulator des ATARI ST noch deutlich verbessern kann, wollen wir Ihnen anhand einer Mini-Serie zeigen. Serie deswegen, weil wir niemand ca. 1800 Programmzeilen in einer Ausgabe zumuten wollen. Doch das fertige Programm (nur 4,5 K lang) rentiert sich, da man Escape-Sequenzen von A - Z und somit viele neue Features erhält (z.B. 2-6 mal schnellere Bildschirmausgabe, erstmalige Grafikfähigkeit für problemlose Übertragung bei DFÜ und vieles mehr). Natürlich bleibt das Programm (abwärts-) kompatibel zum VT 52-Standard.

Seite 122

A L T

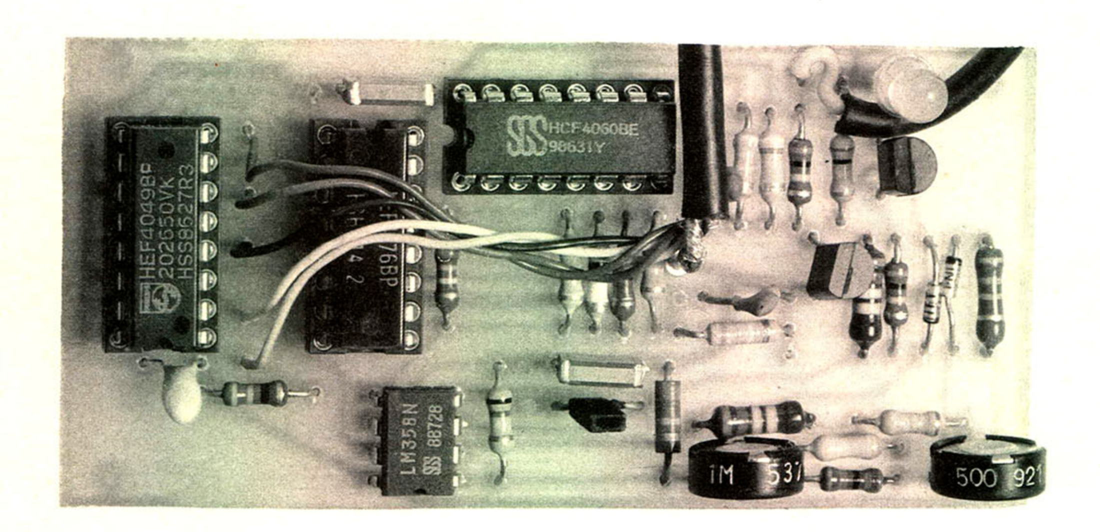
Programmierung relationaler Datenbanken & VIP Professional

"Was nützen die Softwaretest und abgedruckten Listings, wenn man dem Anwender nicht bei der Nutzung der Programme unter die Arme greifen kann?" So kam uns die Idee, eine neue Rubrik in der ST-Computer einzurichten. Ihr Name lautet: Anwendungen. In dieser Ausgabe fangen deshalb zwei Kurse an, die dem Anwender die Arbeit mit seinen Programmen erleichtern sollen. Anfangen wollen wir mit der Programmierung relationaler Datenbanken (z.B. dBMAN oder STandard Base) und dem Einsatz der Tabellenkalkulation VIP Professional. Hierbei werden u.a. kleine Programme und Macros vorgestellt werden.

Relationale Datenbanken Seite 28
VIP Professional Seite 160

ANWENDUNGEN

PROGRAMMIERPRAXIS



Ein Drucker-Scanner im Selbstbau

Nicht jeder kann mit der Maus als Zeichenkünstler glänzen. Allerdings kann jeder mit Hilfe eines Druckers und ein bißchen Elektronik brauchbare Vorlagen in den ATARI ST einlesen bzw. scannen, um es auf neudeutsch zu sagen. Diese Bilder lassen sich dann leicht ein wenig mit einem passenden Zeichenprogramm nacharbeiten und schon erhält man erstaunliche Ergebnisse. Was Sie an elektronischen Bauteilen und als Software benötigen, finden Sie auf **Seite 50**

Betr.: Vorschau aus dem letzten Heft

Im letzten Heft wurde das Software-Paket ST Kontor angekündigt. Der Test wurde auf die Mai-Ausgabe verschoben, da ein weiteres Modul fertiggestellt wurde und dieses auch mit in den Test miteinbezogen werden soll.

Auch der Multisync-Selbstbau hat sich leider verzögert, da wir feststellen mußten, daß es bauliche Unterschiede zwischen den einzelnen ATARI SM124-Monitoren gibt. Der Bericht wird aber sobald wie möglich nachgeholt. Doch es sei gleich vorweg zu sagen, daß sich der Umbau nur erfahrenen Elektronik-Bastlern empfiehlt.

AKTUELLES

RUBRIKEN

NEWS

HDplus - PREISREDUZIERUNG

Die Firma Vortex mit einer Preisreduzierung ihrer höherkapazitiven Hard Disks (30 bis 60 MByte) das Verkaufsübergewicht der 20 MB-Version ausgleichen. So werden dem Kunden 10 MB Mehrkapazität für einen Aufpreis von nur DM 200,- angeboten, so daß die HDPLUS 30 (30 MByte) DM 1498,- gegenüber DM 1138,- für die HDPLUS 20 kostet. Die HDPLUS 60 kostet sogar "nur noch" DM 2398,-, was einem Mehrpreis von DM 150,-/10 Mbyte entspricht. Stichtag für die neuen Endverbraucherpreise ist der 14. März 1988.

Vortex GmbH
Falterstr. 51-53
7101 Flein bei Heilbronn
Tel.: 07131/52061-63

GFA-BASIC 3.0

Gerüchteweise konnte man ja schon einiges über die neue Version des GFA-BASIC hören, doch etwa Anfang Mai wird sie ausgeliefert werden. Zunächst wird wieder nur ein Interpreter verfügbar sein, dem aber schon ein halbes Jahr später der Compiler 3.0 folgen soll.

Das alte GFA-BASIC ist vollkommen neu überarbeitet worden, so daß der neue Interpreter 3.0 deutlich schneller sein wird. Er hat etwa den doppelten Befehlsumfang, was u.a. durch volle Einbindung der GEM-Routinen bewirkt wird. Ferner sind neue C-ähnliche Strukturen implementiert und der Editor verbessert worden. Der Preis beträgt DM 198,-.

GFA Systemtechnik GmbH
Heerdter Sandberg 30
4000 Düsseldorf 11
Tel.: 0211/588011

Neues MODULA-2 Sprachsystem

Unter dem Namen SPC-MODULA-2 ist ein neues Sprachsystem für die ATARI STs mit monochromen Monitor auf den Markt gekommen. Das System ist mit allen seinen Teilen voll in GEM integriert. SPC-MODULA-2 basiert auf einem MODULA-2 Single-Pass-Compiler der ETH-Zürich und implementiert den neuen WIRTH'schen Standard.

Das Sprachsystem enthält:
- einen Compiler mit einer Übersetzungsleistung von bis zu 5000 Zeilen pro Minute
- einen Editor, in dem bis zu 8 Dateien gleichzeitig ediert bzw. gesichtet werden können.
- einen symbolischen Debugger, der bei einem Programmabsturz automatisch aufgerufen wird. Er macht nicht nur die fehlerhafte Stelle im Programmtext sichtbar, er zeigt auch alle im Moment aktiven Moduln und den Zustand der Variablen an.
- eine Shell, die es je nach Speichergröße erlaubt, den Compiler und den Editor resident im Speicher zu halten.
- ein deutsches Handbuch
- eine Einführung in MODULA-2 anhand von SPC-MODULA-2.
- einen Update-Service mit zweimonatig erscheinender Userzeitung

Im Zuge des Update-Service werden neuentwickelte Module zum Selbstkostenpreis abgegeben, z.B. eine generalisierte Window-Bibliothek, Blitter-Einbindung, 68881 Arithmetik, usw.

Eine Demodiskette wird für DM 10,- angeboten. Sie enthält den Compiler mit Editor und Debugger, wobei der erzeugbare Code auf 5K beschränkt ist. Außerdem wird nur das Modul InOut unterstützt.

advanced applications Viczena GmbH
Sperlingweg 19
7500 Karlsruhe 31
Tel.: 0721/700912

Thomas Kaschad Computer-Technik stellt ein neues Buchhaltungsprogramm für Freiberufler und Selbständige vor. TKC-EINNAHME/ÜBERSCHUSS PLUS zeichnet sich laut Hersteller durch hohe Verarbeitungsgeschwindigkeit, Benutzerfreundlichkeit, hohe Absturzsicherheit und günstiges Preis-/Leistungs-Verhältnis aus und stellt somit wesentlich teurere Programme in den Schatten. Für den privaten ST-Anwender wird ab Mitte Februar die Version TKC-HAUSHALT PLUS für DM 129,- angeboten.

TK Computer-Technik
Bischhofsheimer Str. 17
6097 Trebur-Astheim T
el.: 06147/550

Accessory für NEC-Drucker

Den Benutzern der Drucker NEC P6, P7 und P2200 wird von der Buchhandlung Werner Finke Wuppertal ein Accessory zur Einstellung angeboten. Es ist voll auf die Fähigkeiten der NEC-Drucker abgestimmt und beinhaltet u.a. einen Spooler von 32 KBytes. Das NEC P6 ACC ist für einen Preis von DM 29,90 erhältlich.

Buchhandlung Werner Finke
Postfach 131545
5600 Wuppertal 1
Tel.: 0202/454220 oder 454433

Symbolbibliotheken für GFA Draft plus ST und GFA-BASIC-Einsteigerset

GFA Systemtechnik hat bei der Firma S+S Datentechnik einige neue Symbolbibliotheken zum GFA Draft plus ST CAD-Programm gefunden. Die Bibliotheken gibt es für ELEKTROTECHNIK (beinhaltet Starkstrom- und Elektrotechnik), MÖBEL und PNEUMATIK-HYDRAULIK. Eine Bibliothek für GFA Draft plus ST kostet DM 149,-.
Seit 1.3.1988 ist von GFA das GFA-BASIC-Einsteigerset erhältlich. Es beinhaltet den GFA-BASIC-Interpreter, den GFA-BASIC-Compiler, das Programm GFA-VEKTOR und ein Computerlexikon. Das ganze Set ist in einem schwarzen Koffer für DM 198,- zu erwerben und erscheint in limitierter Auflage.

GFA Systemtechnik GmbH
Postfach 190263
4000 Düsseldorf 11

Neues von Tommy Software

Das erste neue TommySoftware-Produkt im Jahr 1988 ist LISP-Complete. Das Paket besteht aus zwei Teilen: einem Lisp-Interpreter mit GEM-Editor, POST-Mortemfunktion und zahlreichen Demonstationsprogrammen und einem Lisp-Lehrgang in Deutsch. Der Lehrgang erläutert ausführlich die Unterschiede und Gemeinsamkeiten mit anderen Programmiersprachen und enthält nach jedem Kapitel Übungsaufgaben.
Das zweite Produkt, das Tommy-Software vorstellt, ist der 1ST Freezer //, der wie sein Vorgänger, der 1ST Freezer, ein Backup-Tool ist. Er ermöglicht als vorgeladenes Programm auf Tastendruck das Einfrieren bzw. Abspeichern eines Programmes. Zusätzlich lassen sich jetzt auch Programme auf doppelseitig formatierten Disketten "freezen". Das Programm läuft auf Farbe und Monochrome und bietet eine etwa 85%-Kompatibilität zu bestehenden Programmen. Zusätzlich ermöglicht der Freezer durch seine Stop-and-Go-Option, Programme für Bildschirmfotos anzuhalten. Der Verkaufspreis von 1ST Freezer // beträgt DM 148,-. Es benötigt mindestens 1MB RAM und TOS in ROM. Es läuft auf allen TOS-Versionen.
Das dritte neue Produkt ist 1ST-Speeder //, der Nachfolger von 1ST Speeder, einem Diskettenbeschleuniger auf Cache-Basis. 1ST-Speeder wurde nochmals beschleunigt und ist jetzt in der Lage, bis zu 3,7 MB RAM verwalten. Außerdem verfügt er über ein abschaltbares Autosetup und ist weiterhin gegen ein nicht funktionierendes Media-Change des Systems gefeit. 1ST-Speeder benötigt mindesten 512 KB RAM, ein Diskettenlaufwerk und läuft auf allen TOS-Versionen und allen Auflösungen. Der Verkaufspreis beträgt DM 118,-.

Für die Programme 1ST-Freezer und 1ST-Speeder bietet Tommy-Software auch einen Updateservice. Gegen Einsendung der Originaldiskette und DM 30,- erhält der Benutzer die neue Version.

Auf der CeBIT '88 wird Tommy-Software drei neue Produkte vorstellen:

1. MegaPaint
Bei MegaPaint korrespondiert ein Bildschirmpixel mit einem Druckerpixel, so daß auch auf einem 9-Nadeldrucker die Qualität eines 24-Nadlers erreicht wird. Weiterhin enthält es proportionale Zeichensätze und einen eigenen Zeichensatzeditor, einen Taschenrechner, und es ermöglicht das Aufarbeiten gescannter Bilder in Druckerpixelgröße. MegaPaint ist DEGAS-kompatibel und erlaubt Textimport. MegaPaint arbeitet mit dem FX-80, P6 und dem ATARI-Laserdrucker zusammen.

2. MultiST
Mit MultiST sind mehrere Applikationen gleichzeitig im Speicher verfügbar. In einem Test gelang es, zehn kleinere Programme in einem normalen 1040er zu starten. Der Verkaufspreis wird etwa DM 100,- betragen.

3. StarTrash
Der Programmierer von Dizzy Wizzard hat wieder zugeschlagen. StarTrash ist so ästhetisch wie Dizzy Wizzard, so actiongeladen wie das berühmte Kugelspiel auf dem Amiga und so faszinierend wie die Sprüche des intergalaktischen Langohrs. Auch dieses Game wird auf der CeBit 88 das erste Mal vorgestellt und weniger als DM 80,- kosten.

Tommy Software
Gutzkowstr. 35
6000 Frankfurt 70
Tel.: 069/614046

C&C
Computers and Communications

P 2200 – DAS PREIS-LEISTUNGS-GENIE

Die Computer-Anwender haben Grund zum Jubeln!

Genial – endlich ein Drucker, der für Einsteiger, Aufsteiger und Semiprofis geeignet und vor allem erschwinglich ist. Denn NEC erschließt Ihnen jetzt die



Vielseitige, durchdachte Papierzuführungen.

modernste 24-Nadel-Drucktechnologie zu einem erstaunlich günstigen Preis.

NEC Pinwriter P 2200 mit automatischer Einzelblattführung

Erleben Sie eine neue Dimension: gestochen scharfen Korrespondenzdruck mit ungewöhnlich reicher Schriftartenauswahl, brillante Grafik-Darstellung, bequeme Druckersteuerung und integrierte Papierzuführungen.

Warum also tief in die Taschen greifen, wenn es schon für wenig Geld 24-Nadel-Technologie mit allen Raffinessen gibt?

PROFIQUALITÄT ZUM AMATEURPREIS

NEC ist mit seinen 24-Nadel-Druckern in Deutschland marktführend.

Was den P 2200 als echten Profi auszeichnet, sind seine hohe Auflösung von 360 x 360 dpi, ein halbes Dutzend serienmäßiger Schriftarten und eine Reihe praktischer Papierzuführungen. Zum Beispiel können Sie zwischendurch einen Brief drucken, ohne daß das Endlospapier extra herausgenommen werden muß.

Durch seine volle Kompatibilität mit den NEC Pinwritern der 24-Nadel-Serie harmoniert der P 2200 mit allen wichtigen Software-Paketen.

EIN NEC DRUCKER FÜR JEDERMANN

Endlich braucht niemand mehr auf die bewährte NEC Produkt- und Druckqualität zu verzichten.

Hervorragende Druckqualität durch bewährte 24-Nadel-Technologie.

Eine breitgefächerte Gruppe – vom Schüler über den Heimanwender bis hin zum Freiberufler – findet im P 2200 die ideale Drucklösung. Anwendern, die schon seit langem auf der Suche nach einem preisgünstigen Drucker für ihren



12 Schriftartenkassetten zusätzlich erhältlich.

Computer sind, eröffnet er die Möglichkeit, Druckergebnisse in bewährter NEC-Qualität zu erzielen.

Damit ist der P 2200 die optimale wirtschaftliche und technische Alternative für alle, die sich bei gleichem finanziellen Aufwand bisher nur mit antiquierten 9-Nadel-Druckern begnügen mußten.

Für Beratung und Service stehen Ihnen Ihr NEC Drucker-Fachhändler und sein Team zur Verfügung.



Weitere Informationen erhalten Sie von:

NEC Deutschland GmbH
Klausenburger Straße 4, 8000 München 80
Tel.: 0 89/9 30 06-0, Telefax: 0 89/93 77 76/8
Telex: 5 218 073 und 5 218 074 nec m

C&C
Computers and Communications

NEC PINWRITER UND TEXTVERARBEITUNG

12 Zeichen pro Zoll i
20 Zeichen pro Zoll im Kleindruck -
17 Zeichen pro Zoll im Kleindru
15 Zeichen pro Zoll im Schr
12 Zeichen pro Zoll i
10 Zeichen pro Zol
20 Zeichen pro Zoll in Briefqualitä
17 Zeichen pro Zoll in Briefqu
15 Zeichen pro Zoll in Bri
12 Zeichen pro Zoll i
10 Zeichen pro Zol

Verschiedene Zeichen ...

dreifach

chrift, unterstrichener
ware kann man auc
ppeltes Unterstreichen
Pinwriter P5 XL, bi
ett, Gelb, Orange, Grü



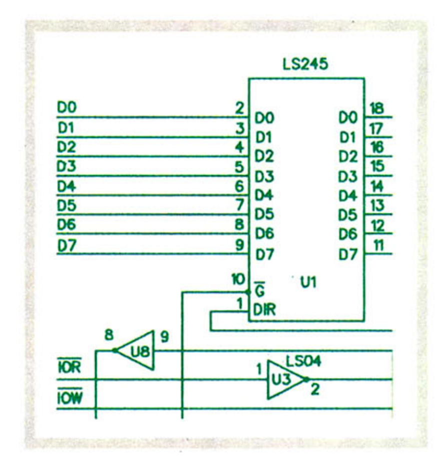
geh
n diesem Beispiel

... in vielen Kombinationen ...

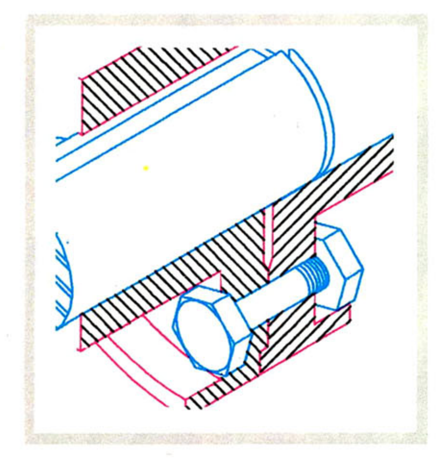


... bilden perfekte Texte.

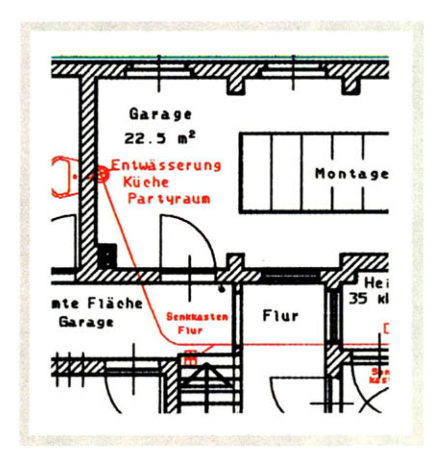
NEC PINWRITER UND CAD



Höchste Druckpräzision, ...

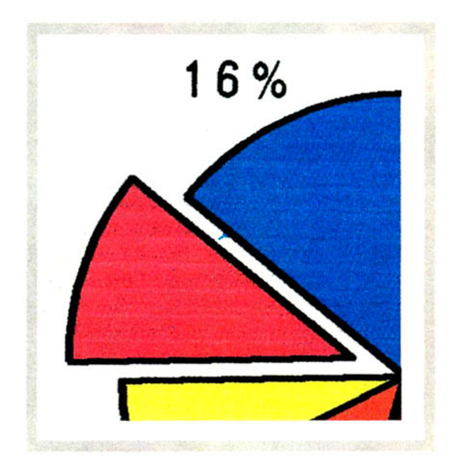


... kombiniert mit Farbe, ...

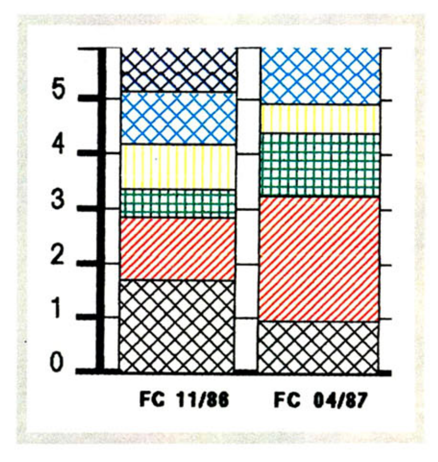


... liefert korrekte Pläne.

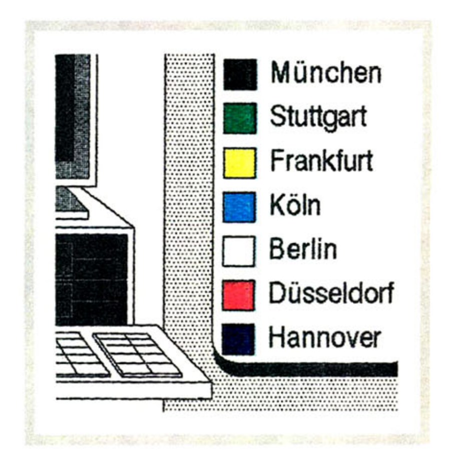
NEC PINWRITER UND BUSINESS GRAFIK



Mit vielen Farben ...



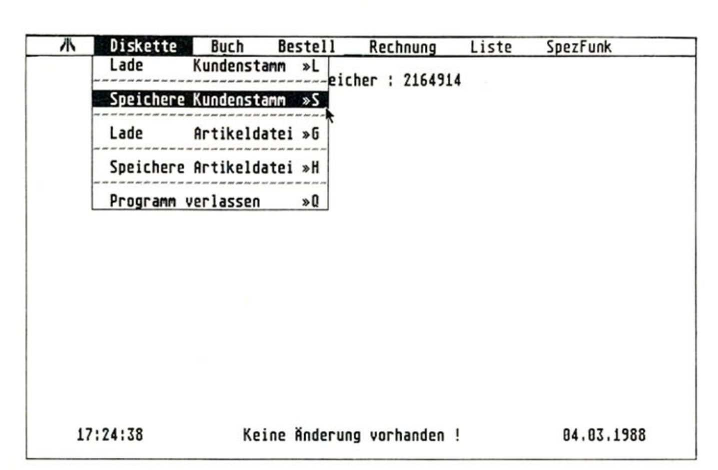
... in allen Schattierungen ...



... gelingt jede Präsentation.

Alle Abbildungen sind im Maßstab 1:1 unretuschiert dargestellt und wurden mit NEC Pinwritern ausgedruckt.

MEGAwork ST



Aus Bottrop kommt ein Programm namens MEGAwork ST, das unter dem Motto "Software zum Arbeiten, Überflüssiges stört nicht" angeboten wird. Es ist als Arbeitsprogramm für Klein- und Mittelständische Betriebe sowie freiberufliche Computeranwender gedacht.

Es enthält eine Kundenverwaltung von bis zu 2500 Kunden, wobei Anschrift, Bestellung und Hinweistexte zu jedem Kunden möglich sind. Ebenso ist eine halbautomatische Abhandlung von Bestellungen und Rechnungen etc. integriert. Dazu lassen sich bis zu 1500 Artikel mit dem Programm erfassen.

Ferner besteht die Möglichkeit, Serienbriefe über einen individuellen ASCII-Editor einzugeben, wobei die Formulare beliebig aufgebaut werden können. Es sind sogar verschiedene Kalkulationsmöglichkeiten gegeben.

Natürlich lassen sich Kunden und Artikel nach beliebigen Kriterien suchen und Listen auf Drucker, Monitor oder Diskette ausgeben. Ebenfalls lassen sich Adressaufkleber drucken u.v.m.

Benötigt wird ein ATARI ST mit mindestens 1 MB RAM und TOS im ROM, ein Farb- oder Monochrom-Monitor, ein doppelseitiges Laufwerk und ein Drucker. Der Preis beträgt DM 198,-.

Mega Team
Kirchhellener Str. 262
4250 Bottrop
Tel.: 02041/94842

Datenbank von COMPUTERWARE

Auf der CeBIT '88 wird die Firma COMPUTERWARE Gerd Sender die neuen Programmpakete Regent Base 2.0 (eine profesionelle Datenbank mit SQL) und The Informer (eine flexible Datenbank unter GEM) erstmals der breiten Öffentlichkeit vorstellen. Die funktionelle Textverarbeitung Regent Word II, mit der Möglichkeit, Serienbriefe zu schreiben, wird in beiden Programmpaketen (ohne Aufpreis!) enthalten sein. Regent Base 2.0 soll DM 349,- und The Informer DM 249,- kosten. Außerdem sind ein SQL-Extender für GFA-BASIC, der einem die leistungsfähige Datenbanksprache SQL mit ihrem umfangreichen und flexiblen "Wortschatz" in GFA-BASIC zugänglich macht (Die erstellten Datenbanken sind kompatibel mit Regent Base 2.0), und ein Hard Disk-Repair-Programm von Beckemayer Development Tools zur Erkennung und Behebung von Fehlern für die Harddisk am ATARI ST für dieses Frühjahr angekündigt worden.

Computerware Gerd Sender
Moselstr. 39
5000 Köln 50

Neuer Drucker von Star



Der "Einstiegs-Drucker" NL-10 von Star, von dem in den letzten zwei Jahren mehr als 300.000 Stück verkauft wurden, wird durch den LC-10 abgelöst. Er zeichnet sich durch mehr Schriften, mehr Bedienungskomfort und weniger Lärm aus.

2 MByte RAM Upgrade

Die Firma Weide Elektronik GmbH bietet eine RAM-Erweiterung auf 2 MBytes für alle ATARI ST mit 1 MByte RAM an. Normalerweise wird dazu eine Speicherbank mit 1 MBit-RAMs bestückt. Bei dieser Speichererweiterung werden 256 KBit-RAMs verwendet, die auch schon im ATARI ST mit 1 MByte RAM vorhanden sind. Es wird keine zusätzliche Software benötigt, da der Speicher von der MMU mitverwaltet wird. Die Erweiterung findet im Rechner Platz und erfordert nur 6 Lötstellen. Eine Version, die keine Lötarbeiten mehr benötigt, ist bereits in Arbeit und soll voraussichtlich Mitte März lieferbar sein.

Der Preis für die Speichererweiterung beträgt DM 598,-.

Weide Elektronik GmbH
Regerstr. 34
4010 Hilden
Tel.: 02103/41226

Neues von UVS

Das Universal-Lernprogramm Lern ST wird in einer verbesserten Version für DM 59,- angeboten. Dazu gibt es erstmals zwei Sprachdisketten für Latein und Englisch zu einem Preis von je DM 20,-.

Seit kurzem erscheint regelmäßig alle drei Monate die Public Domain-Diskettenzeitschrift CIP ST. Man kann sie zu einem Preis von DM 30,- (4 Ausgaben) im Jahresabo beziehen.

Als Ergänzung zum GFA-BASIC-Compiler bietet UVS zum erstenmal einen Linker an. Link_it ermöglicht das beliebige Dazulinken von Assembler-Files und compilierten Programmen, so daß das lästige Nachladen von Maschinenspracheroutinen entfällt. Link_it kostet DM 29,- inkl. Routinenbibliothek.

Ein anderes Zusatzprodukt zum GFA-BASIC heißt 1st BASIC-Tool und kostet DM 20,-. Das 1st BASIC-Tool enthält u.a. einen verbesserten Input-Befehl und eine neue Fileselector-Box. Die Routinen dürfen lizenzfrei in eigener Software verwendet werden.

Zu den beiden Software-Paketen Adress ST/Check ST (Adreßverwaltung und Textkorrekturprogramm) und Label ST (Diskettenaufkleberbedruckprogamm und Disk-Copy-Programm) werden neuerdings die Quellcodes mitgeliefert, damit der Anwender aus den verwendeten Routinen noch lernen kann. Der Preis der beiden Programme liegt bei je DM 29,-.

Ulrich Veigel Softwareservice (UVS)
Mönchseestr. 83-85
7100 Heilbronn
Tel.: 07131/81781

Interface für Casio FX-850P

Um den ATARI STmit dem Casio FX-850P zu koppeln, wird jetzt ein Interface angeboten. Es bietet eine bidirektionale Datenübertragung mit max. 1200 Baud. Erfreulich ist, daß kein externes Netzteil benötigt wird. Geliefert wird es komplett anschlußfertig und betriebsbereit mit Software für den ST. Der Preis beträgt DM 99,-.

Jochen Flimm / Markus Wolf
Hardware / Software
Bodelschwinghweg 15
6300 Giessen
Tel.: 0641/34232

GFA-BASIC Linker

Aus den Niederlanden wird ein Linker für GFA-BASIC angeboten. Er ermöglicht die Anlage von Modul-Bibliotheken (z.B. AES- und VDI-Routinen) und bindet die benötigten Module dann ins Programm ein. Ferner wird ein Programm zur Zeilennumerierung und zur Erstellung einer Cross-Reference-Liste mitgeliefert. Die Gebrauchsanleitung ist in deutsch. Der Preis beträgt DM 25,-.
Von der gleichen Firma wird ein Codierprogramm namens ST-Coder angeboten. Mit ihm läßt sich jede Art von Datei gegen Einsicht von Unbefugten schützen. Es setzt die Daten in eine völlig unlesbare Form um. Der Schlüssel zum Decodieren ist frei-wählbar und wird nicht auf Diskette geschrieben. Die codierte Datei wird Byte für Byte über die ursprüngliche Datei abgespeichert, so daß auch mit einem Disketten-Monitor kein unbefugter Zugriff stattfinden kann. Der Preis beträgt ebenfalls DM 25,-.

HiQ Systems
Postfach 21766
NL-3001 AT Rotterdam

ST-BASIC-Compiler

Der neue Jam Basic Compiler lehnt sich ans ST-Basic an. Er unterstützt ca. 60 Befehle sowie die Datentypen string und integer. Die compilierten Programme laufen bis zu 10 mal schneller als im Interpreter. Ferner ermöglichen die zu erweiternden Bibliotheken einen bequemen Umgang mit TOS und GEM. Der Compiler kostet DM 20,- und ist erhältlich bei:

Christoph Jakfeld
Alsfeldstr. 23
4200 Oberhausen 11

Neuer Scanner FS 2

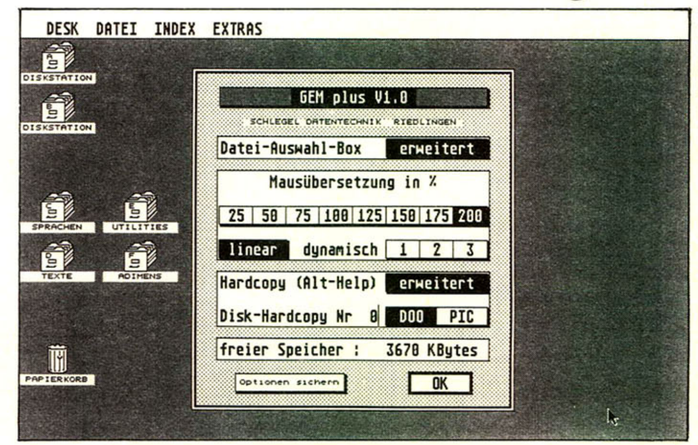
Ein neuer Flachbett-Scanner wird von Wilhelm Mikroelektronik angeboten. Er ermöglicht ein monochromes Digitalisieren durch einen CCD-Sensor. Der Scanner funktioniert ohne bewegten Schlitten, so daß auch sperrige Objekte verarbeitet werden können. Die Features des Scanners lauten:

- Größtmöglicher Scan-Bereich : 21,6 x 35,6 cm
- Scan-Geschwindigkeit : 10 s für DIN A4 bei 300 dpi
- Auflösungen : 200, 300, 400 dpi - 16 Grauwerte
- Pseudo-Graustufenerzeugung durch Graumuster (Raster)
- Realtime-Verkleinerung während des Scan-Vorgangs auf 100, 150 und 200 dpi ohne Geschwindigkeitseinbußen.
- Scanbereich voreinstellbar
- Fotokopierfunktion auf dem ATARI Laserdrucker (1:1); dabei stufenlose Verkleinerung oder Vergrößerung.
- Vollständig durchgeschleifte gepufferte DMA-Schnittstelle nach ATARI-Norm

Im Lieferumfang ist ein sehr umfangreiches Malprogramm zur Bearbeitung der Grafiken enthalten. Zusätzlich ist zur Zeit ein Softwaremodul zur optischen Zeichenerkennung (es werden ca. 200 Schriften erkannt), eine Option für FAX-Empfang- und Sendevorgänge und Entwicklerunterlagen zur Einbindung des Scanners in eigene Programme lieferbar bzw. in Vorbereitung.

Wilhelm Mikroelektronik
Süggelstr. 31
4670 Lünen
Tel.: 02306/25299

GEMplus-Accessory



Ein Utility-Accessory für den ATARI ST bietet Schlegel Datentechnik an. Es beinhaltet eine erweiterte Fileselektorbox, bei der bis zu acht Laufwerke auf Knopfdruck auswählbar sind. Ferner sind vier Extension-Wahlknöpfe, von denen einer frei definerbar ist, vorhanden. Sie besitzt ein Schnell-Schließfeld, mit dem man sofort in die oberste Directory-Ebene kommt usw.
Als weiteren Menüpunkt kann man beim GEMplus-Accessory die Mausgeschwindigkeit einstellen, optional auch dynamisch wie beim Apple MacIntosh.
Es lassen sich mit GEMplus auch Hardcopies (quer und normal) und Screendumps (Doodle-Format) erzeugen. Ferner wird der freie Speicherplatz ausgegeben. Zu guter letzt lassen sich alle eingestellten Parameter abspeichern. Der Preis beträgt DM 59,-.

Schlegel Datentechnik
Schwarzachstr. 3
7940 Riedlingen
Tel.: 07371/2317

SCHÖNERE PROGRAMME

GEM-Fonts unter

GEM bietet einige sehr interessante Möglichkeiten, die auf dem ST leider noch gar nicht richtig ausgenutzt werden. Ein wichtiger Punkt dabei sind die Textdarstellungen. Da jeder Text im Grafikmodus ausgegeben wird, bietet GEM (VDI) die Möglichkeit, verschiedene Textattribute (fett usw.) einzustellen und die Textgröße zu wählen. Davon machen ja auch schon viele Programme (v.a. im Grafikbereich) reichlich Gebrauch. Die Entwickler von GEM sind aber noch einen sehr großen Schritt weitergegangen. Man kann außer den System zeichensätzen noch andere Fonts von Diskette laden (sollte man wenigstens können). Das Format dabei ist außerdem recht flexibel (Proportionalschrift usw.). Dabei tritt nur ein großes Problem auf: die Implementation auf dem Atari ST. Diese ist nämlich nicht ganz vollständig, bzw. fehlerfrei. Mit dem normalen, nicht erweiterten Betriebssystem ist es deshalb nicht möglich, diese Zeichensätze zu benutzen.



Aus diesem Grund gibt es prinzipiell zwei Möglichkeiten, auf dem ST-Zeichensätze zu laden. Die eine ist sozusagen die offizielle: Man packt das Programm GDOS.PRG (gehört nicht zum Lieferumfang des ST) in den Autoordner der Programm- bzw. Systemdiskette, muß alle einzelnen Fontgrößen auf diese Diskette kopieren und die möglichen Fonts vorher in einer Datei namens ASSIGN.SYS festlegen. Dann, und nur dann, kann das eigentliche Programm die VDI-Funktionen vst_load_fonts und vst_unload_fonts (119 und 120) benutzen, da sie dann funktionieren. Dieses Verfahren kann man natürlich anwenden, es hat aber im Vergleich zu dem folgenden sehr viele Nachteile: der Anwender muß neu booten (wer bootet schon immer mit GDOS.PRG und dem richtigen ASSIGN.SYS), die Zeichensätze sind vorher festgelegt (es kann nicht im Programm gewählt werden), und es schwirren sehr viele Dateien im Hauptdirectory (zwingend!) der Diskette rum, für jede Größe eine. Deshalb sollte man sich bei Interesse auf jeden Fall mit folgendem Verfahren näher beschäftigen: Die Zeichensatzdateien werden direkt in den Speicher geladen. Sie haben ja einen definierten Aufbau und auch die interne Organisation der Fonts ist dokumentiert. Dies ist gar nicht so schwierig, wie es zunächst erscheinen mag, allerdings auch nicht ganz einfach.

Im folgenden beschreibe ich eine Lösung für GEM-Programme unter ST-Pascal plus. Die Anwendung ist recht einfach, da es sich lediglich um zwei Include-Dateien handelt, die einige Funktionen zur Zeichensatzanwendung zur Verfügung stellen.

Um das Prinzip zu verstehen, sollte man sich zunächst den Aufbau von Font-Dateien verdeutlichen. Jeder Zeichensatz besteht prinzipiell aus zwei Teilen, dem Font-Header und den eigentlichen Zeichensatzdaten und einer (oder zwei) Tabellen. Für die Lösung unseres Problems ist vor allem der Font-Header interessant. Für Pascal läßt er sich als Record darstellen, wie es in FONTTYPE.PAS (Listing 1) geschehen ist. Sehr wichtig ist der Eintrag NextFont, da die geladenen Fonts im Speicher über diesen Zeiger verkettet sind. Die anderen Einträge sind auch wichtig, allerdings verweise ich für genauere Informat unten angegebenen Publikationen, vor allem auf das zweite Sonderheft der ST-Computer. Außerdem sollten Sie sich einmal das Listing 1 ansehen, da ich im folgenden von Datentypen spreche, die dort definiert sind.

Das Prinzip beim Laden eines Fonts besteht darin, vom Betriebssystem entsprechend Speicher anzufordern, die Datei dorthin zu laden, und die Zeiger entsprechend zu korrigieren (wie es so schön heißt, den Zeichensatz in die Liste einzuhängen). Ganz so einfach ist es allerdings nicht, wie ich weiter unten zeigen werden. Doch wo, werden sich findige Leser jetzt fragen, soll ich den ersten Zeichensatz denn "einhängen"? Dazu muß man das eigentliche GEM verlassen und die Line-A-Routinen betrachten, die für die Ausgabe von Grafik (und auch Text) auf den Bildschirm zuständig sind. Die Funktion Line-A-



DURCH ZEICHENSÄTZE

ST-pascal plus.



Init ($A000) gibt drei Adressen zurück, wovon eine auf eine Tabelle zeigt, die wiederum die Adressen der Systemfont-Header enthält (Kommen Sie noch mit?). Aus dieser Tatsache ergibt sich auch die Aufgabe einer Prozedur in meinen Routinen, nämlich von FirstFontPtr. Diese Funktion liefert einen Zeiger auf den Header des ersten GEM-Fonts zurück (Datentyp FontPtr oder auch Font). Wie oben schon erwähnt, ist dazu allerdings etwas Assembler notwendig. Der Einfachheit halber habe ich diese kurze Routine in ein Feld gepackt und zum Aufruf die XBios-Routine SupExec (38) benutzt. Der Aufruf erfolgt zwar im Supervisormodus, und man darf keine Betriebssystemfunktionen benutzen, aber das stört hier nicht (dieses Verfahren bietet sich sowieso bei kleinen Routinen an, wenn man nicht gleich eine extra Objekt-Datei benutzen will.) Nach Aufruf dieser Funktion steht in Routine. Pointer die Adresse des ersten GEM-Fonts (gleich als richtiger Typ). Für Interessierte: Der Quelltext der Routine steht in FONTINIT.ASM (Listing 2b). LastFontPtr benutzt FirstFontPtr und "hangelt" sich dann die Liste entlang, um den letzen Font zu bestimmen. Dort werden neue Fonts angehängt.



Um einen Zeichensatz zu laden (FNT-Datei), wird die Funktion LoadFont benutzt. Diese öffnet die Datei und ruft dann InstallFont auf. InstallFont erledigt die eigentliche Arbeit. Ich möchte das eigentliche Funktionsprinzip nur kurz beschreiben, sehen Sie sich das Listing an, oder benutzen Sie die angegebene Literatur.

Zuerst wird entsprechend der Dateilänge vom Betriebssystem Speicher angefordert. Aus diesem Grund muß bei Benutzung von FONTSUBS folgendes auf jedem Fall gewährleistet sein: Das Betriebssystem muß genügend Speicher haben! Also am besten die U-Option des Compilers benutzen; {$U100} wird wohl reichen. Nachdem also hoffentlich Speicher reserviert werden konnte, wird die Datei an diese Stelle mit Hilfe von FRead (Gemdos) eingelesen. Nun folgen aber noch einige Operationen, die vorallem folgenden Sinn haben: Da GEM auch auf PCs zu finden ist, sind im Fontheader und in den Tabellen Worte im Intel-Format abge speichert (erst Low-Byte, dann Hi-Byte). Im Speicher müssen sie aber richtig vorliegen (der ST benutzt einen MC68000). Aus diesem Grund werden im größten Teil des Fontheaders und in den dann folgenden Tabellen die Worte "motorolatisiert". Die eigentlichen Zeichensatzdaten sind aber schon richtig vorhanden (wenn auf dem ST erstellt). Nach diesen Vorgängen werden die Offsets noch …dressen erweitert und der Zeichensatz in die …iste "eingehängt". InstallFont gibt dann den Zeiger auf den Zeichensatz als FontPtr zurück. Die entgegengesetzte Funktion zu LoadFont ist UnloadFont. Der Zeichensatz wird aus der Liste "ausgehängt" und der belegte Speicher wieder freigegeben.

Mit diesen beiden Funktionen können Sie schon mehr machen als mit den Original-Funktionen. Wenn Sie einen Zeichensatz laden wollen, müssen Sie folgendermaßen vorgehen: Sie fügen in GEM-Programmen (und nur dort) hinter {$IGEMTYPE} bzw. {$IGEMSUBS} {$IFONTTYPE} und {$IFONTSUBS} ein und deklarieren dann eine Variable vom Typ Font. Um dann einen Zeichensatz zu laden, machen Sie folgendes (Ihre Variable heißt z.B. Zeichens):

```
Zeichens:=LoadFont ('DATEI NAM.FNT',2);
```

'DATEI NAM.FNT' steht dabei für den Namen der Fontdatei, die 2 für die Nummer, unter der der Font angesprochen werden soll. Wenn Zeichens danach NIL ist, ist …hler aufgetreten …nicht vorhanden …enig Speicher).



Wenn Sie nun diese Schrift benötigen, kommt Text_Face zum Einsatz: Text_Face(2); wählt den Font mit der Nummer zwei. Alle Grafik-Text-Ausgaben benutzen dann diesen Font bis Sie mit Text_ Face(1); wieder den Systemfont einschalten. So einfach kann die Anwendung von Zeichensätzen sein (oder ist das etwa nicht einfach ?!). Vor Programmende müssen Sie dann nur noch Unload Font(Zeichens); aufrufen, um den Speicher wieder ordnungsgemäß freizugeben.

Bis hierhin ist es ja schon ganz schön und ich war auch ganz erfreut, daß ich nach einigen Mühen und leider auch Abstürzen zum ersten Mal Proportionalschrift auf dem "Pascal-Bildschirm" sah. Es bleibt nur noch ein Problem: die vielen Dateien auf den Disketten. Eine eigene Datei für jede Größe jedes Zeichensatzes ist nun wirklich nicht nötig. Es sieht nicht nur unübersichtlich aus, sondern benötigt effektiv auch mehr Platz auf der Diskette und mehr Ladezeit. Deshalb habe ich mir ein Format ausgedacht, um mehrere Dateien sinnvoll zusammenzufassen. Es ist zugegebenermaßen nicht sehr aufwendig, aber dafür extrem nützlich. Hier nun eine Beschreibung dieses Formats:

Die ersten zwölf Bytes stellen eine Kennung da, um die Datei zu identifizieren. Sie müssen "ZSA-GEM-FONT" enthalten, da ich solch eine Datei ZSA-Datei nenne (von Zeichensatz, klar?).

Das nächste Wort gibt nun die Anzahl der verschiedenen Fontarten (Faces) an. Für jedes Face wiederholen sich die folgenden Daten:

1 Wort: Anzahl der Zeichensatzgrößen für jedes Face.

Für diese einzelnen Größen wiederholen sich nun wiederum die nächsten Daten:

1 Wort: Länge der eigentlichen Zeichensatzdatei (FNT).

danach: Die Daten dieser FNT-Datei ohne jede Änderung.

Wenn man eine so aufgebaute Datei laden will, kommen andere Funktionen zum Einsatz. Sie brauchen keine Variable des Types Font mehr, sondern dafür eine vom Typ FontList. Der Funktion LoadZsaFonts übergeben Sie folgende Parameter: Zuerst wieder den Dateinamen (*.ZSA), dann die Face-Nummer; diese Face-Nummer gilt für den ersten geladenen Face. Alle weiteren Faces werden fortlaufend numeriert. Der letze Parameter ist die eben erwähnte "Font-Liste". Die Funktion LoadZsaFonts lädt alle Fonts in den Speicher. Das genaue Vorgehen entnehmen Sie bitte dem Listing. Es wird auch InstallFont benutzt, die Hauptsache funktioniert also genauso wie bei LoadFont. LoadZsaFonts gibt die Anzahl der korrekt geladenden Faces zurück. Bei Auftreten eines Fehlers ist der Funktionswert Null.

Die nachgeladenen Fonts können Sie dann mit Text_Face(2,3,4 usw.) ansprechen. Wenn das Programm mit seiner Arbeit fertig ist, muß es nicht UnloadFont sondern UnloadFonts (das kleine s ist wichtig!) aufrufen. UnloadFonts verlangt genau wie UnloadFont einen Parameter, aber nicht vom Typ Font, sondern vom Typ FontList. Es werden alle Zeichensätze freigegeben, und es kann auf sie nicht mehr zugegriffen werden.

Mehrere Zeichensätze in einer Datei

Meiner Meinung nach sind diese zusammengefaßten Zeichensätze in den meisten Fällen eine große Erleichterung. Um nun auch das Erstellen einer solchen Datei zu erleichtern, habe ich ein kleines GFA-Basic-Programm geschrieben (Listing 4). Es ist zwar nicht sehr komfortabel, aber trotzdem ist die Bedienung recht einfach, so daß ich hier nicht weiter darauf eingehe.

Das Anwenden von GEM-Fonts reduziert sich durch diese Routine auf folgende Schritte: Include-Files an den richtigen Stellen definieren. LoadFont oder LoadZsaFonts mit den entsprechenden Variablen aufrufen. Bei Bedarf mit Text_Face die gewünschte Schriftart bestimmen und am Ende UnloadFont oder UnloadFonts aufrufen.

Einen Hinweis muß ich allerdings noch anbringen: In dieser Form laufen die Routinen nur mit der neuesten Version von Pascal plus. Ich benutze z.B. manchmal Long_Integers als Schleifenvariablen, und außerdem sind in älteren Versionen einige GEM-Funktionen nicht vordefiniert. Ein Umschreiben müßte möglich sein, doch empfehle ich sowieso auf jeden Fall, sich die neueste Version zu besorgen (Das Update ist gar nicht so teuer, es lohnt sich.).

Ab jetzt steht also der Verschönerung von GEM-Programmen durch verschiedene Zeichensätze nichts mehr im Wege. (Außer daß Sie keine Zeichensätze haben?! Da kann ich Ihnen auch nicht helfen. Aber trösten Sie sich, ich habe da auch meine Probleme. Man bekommt wirklich fast keine. Aber vielleicht ändert sich das jetzt; als Anregung: Wer schreibt einen guten Font-Editor?)

Konvertierung der Zeichensätze

Es gibt allerdings eine Lösung für dieses Problem der fehlenden GEM-Zeichensätze: Die Konvertierung von anderen Formaten. Sicherlich gibt es einige Leser, die STAD- oder Fontloader-Fonts haben. Solche gibt es teilweise auch schon auf Public-Domain-Disketten. Nun habe ich ein kleines GFA-Basic-Programm geschrieben, das diese Konvertierung vornimmt. Ein Nachteil ist, daß solche Zeichensätze nicht proportional und nur in der Standardgröße vorhanden sind. Trotzdem lohnt es sich, sie zu benutzen. Es ist immerhin etwas.

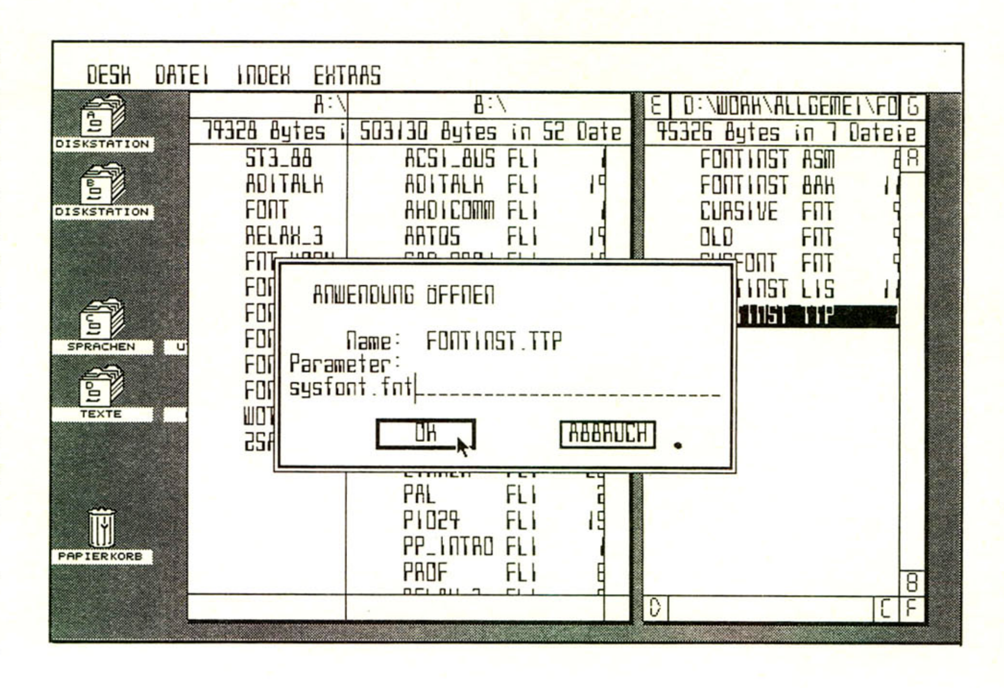
Zur Anwendung des Programms (Listing 4) ist folgendes zu sagen:

In einer Alert-Box wählen Sie die Art des zu konvertierenden Fonts, dann den Namen. Danach wird er in den Speicher geladen und Sie werden aufgefordert, den Namen einzugeben, der in den Font-Header eingetragen wird. Danach geben Sie den Dateinamen des GEM-Fonts ein und er wird auf Diskette geschrieben. Dazu noch eine Anmerkung: Lassen Sie sich ein System einfallen, um GEM-Fonts von anderen Fonts zu unterscheiden, da alle die Extension "FNT" haben.

geht weiter...

Nun noch einige Hinweise zu dem beigefügten Beispielprogramm (Listing 3): Es macht nicht besonders viel: Sie können eine FNT- oder ZSA-Datei wählen und dann Face, Attribute und Größe einstellen. Nach diesen Parametern wird dann ein kleiner Text ausgegeben. Vertiefen Sie sich mal in das Listing, da das Programm sonst keinen praktischen Wert hat.
Noch viel Spaß beim Programmieren, vorallem mit dem ausgezeichneten Pascal plus, und vielleicht erstellt ja jemand mal Zeichensätze. Ich würde mich darüber freuen.

Lutz Preßler



Weitere Informationen finden Sie in:

1) GFA-Basic-Buch (GFA) vor allem im Kapitel Fonts (dort ist eine ähnliche Lösung für GFA-Basic beschrieben).
2) Atari ST - GEM (Data Becker)
3) Atari ST Intern (Data Becker)
4) ST-Computer-Sonderheft 2 (Heim-Verlag)
außerdem natürlich auch noch in anderen Büchern über GEM.


```
1:  { Listing 1: }
2:  { ========== }
3:
4:  { FONTTYPE.PAS  -  Typendefinitionen für FONTSUBS.PAS
5:                     GEM-Fonts unter ST-Pascal plus
                         laden.   LPSoft 1987 }
6:
7:         FontPtr     = ^FontHeader;
8:         FontHeader  = Record
9:                         FontId          : Integer;
10:                        FontSize        : Integer;
11:                        FaceName        : Packed Array
                              [1..32] of Char;
12:                        LowADE          : Integer;
13:                        HighADE         : Integer;
14:                        TopLine         : Integer;
15:                        AscentLine      : Integer;
16:                        HalfLine        : Integer;
17:                        DescentLine     : Integer;
18:                        ButtomLine      : Integer;
19:                        CharWidth       : Integer;
20:                        CellWidth       : Integer;
21:                        LeftOffset      : Integer;
22:                        RightOffset     : Integer;
23:                        Thickening      : Integer;
24:                        UnderLineSize   : Integer;
25:                        LightMask       : Integer;
26:                        SkewMask        : Integer;
27:                        Flags           : Integer;
28:                        HorOffsetTable  : Long_Integer;
29:                        CharOffsetTable : Long_Integer;
30:                        FontData        : Long_Integer;
31:                        FormWidth       : Integer;
32:                        FormHeight      : Integer;
33:                        NextFont        : FontPtr;
34:                      End;
35:        Font        = FontPtr;
36:        FontListPtr = ^FontListEle;
37:        FontListEle = Record
38:                        Pointer       : FontPtr;
39:                        Next          : FontListPtr;
40:                      End;
41:        FontList    = FontListPtr;
42:
43: { END OF FONTTYPE.PAS }
```

```
1:  { Listing 2: }
2:  { ========== }
3:
4:  { FONTSUBS.PAS  -  'GEM-Fonts' unter ST-Pascal plus
5:    laden und benutzen. 1987 by LPSoft Lutz Preßler,
6:      2904 Hatten    23.11.87 }

7:  { ************* VORHANDENE FUNKTIONEN  ****************
8:   Function  LoadFont(<Name:String>,<FaceNo:Integer>):
                          <FontPtr>;
9:  -> Eine Zeichensatzdatei laden (<Name>:*.FNT) und als
         Face <FaceNo>
10:       installieren. Rückgabewert (FontPtr): Zeiger auf
11:       Zeichensatz oder NIL bei Fehler.
12:  Function  LoadZsaFonts(<Name:String>,<FaceNo:Integer>,
13:       <List:FontList>) : <Integer>
14: -> Mehrere Zeichensätze im ZSA-Format laden und die
          darin enthalten Faces ab Nummer <FaceNo>
15:       installieren. <Name>:*.ZSA, <List>: Liste von
16:       Font-Pointern (für späteren Zugriff und Löschen;
         muß nicht leer sein).
17:       Rückgabewert (Integer): Anzahl der geladenen Faces
             oder 0 bei Fehler.
18:  Procedure  UnloadFont(<Ptr:FontPtr>)
19: -> Einen Zeichensatz (Ptr) löschen. Er kann danach
20:       nicht mehr benutzt werden, der belegte Speicher
          wird freigegeben.
21:  Procedure  UnloadFonts(<List:FontList>)
22: -> Mehrere Zeichensätze löschen (Liste der Zeichens.:
23:       List). Siehe UnloadFont. Die Liste ist danach leer
         (NIL).
24:  Procedure  Text_Face(<FaceNo:Integer>)
25: -> Textface auswählen. Bei nicht vorhandenem Face
26:       wird der Systemfont benutzt.
27: Die anderen Funktionen werden eigentlich nur intern
         benötigt.
28: Anm.: Der Datentyp Font entspricht FontPtr.
29:     ********************************************** }
30:
31:  Function  FontAdress(Ptr:FontPtr): Long_Integer;
32:  Var X: Record Case Integer Of
33:          0: (Adr: Long_Integer); 1: (Ptr: FontPtr); End;
34: Begin {$P-} X.Ptr:=Ptr; FontAdress:=X.Adr;
       {$P=} End {of FontAdress};
35:  Function  FontPointer(Adr:Long_Integer): FontPtr;
36:  Var X: Record Case Integer Of
37:          0: (Adr: Long_Integer); 1: (Ptr: FontPtr); End;
38: Begin {$P-} X.Adr:=Adr; FontPointer:=X.Ptr;
       {$P=} End {of FontPointer};
39: Function FirstFontPtr: FontPtr;
40:  Type RoutType = Record
41:                  Code:     Array[1..7] of Long_Integer;
42:                  Pointer: FontPtr;
43:                End;
44:  Var  Routine: RoutType;
```

geht weiter...

```
 45:   Procedure CallSup(Var Code:RoutType); XBios(38);
 46:  Begin
 47:     Routine.Code[1]:=$48E780E0;
           Routine.Code[2]:=$A0002269;
 48:     Routine.Code[3]:=$00042269;
           Routine.Code[4]:=$005441FA;
 49:     Routine.Code[5]:=$000C2089;
           Routine.Code[6]:=$4CDF0701;
 50:     Routine.Code[7]:=$4E754E71;
 51:     CallSup(Routine);
 52:     FirstFontPtr:=Routine.Pointer;
 53:  End {of FirstFontPtr};
 54:  Function LastFontPtr: FontPtr;
 55:   Var P: FontPtr;
 56:  Begin
 57:     {$P-} P:=FirstFontPtr;
           While P^.NextFont<>NIL do P:=P^.NextFont;
 58:     LastFontPtr:=P; {$P=}
 59:  End {of LastFontPtr};
 60:   Function InstallFont(FaceNo,FHandle:Integer;
          Flang:Long_Integer): FontPtr;
 61:   Var Adr,Anz,I : Long_Integer;
 62:       Ptr,PrPtr : FontPtr;
 63:    Function SwapWord(X:Integer): Integer;
 64:    Begin SwapWord:=Shl((X&$FF),8)+(Shr(X,8)&$FF); End;
 65:    Function Malloc(Amount:Long_Integer): Long_Integer;
         Gemdos($48);
 66:    Procedure Fread(H:Integer; C,Buf:Long_Integer);
         Gemdos($3F);
 67:    Function PeekWord(Adr:Long_Integer): Integer;
 68:    Var X: Record Case Integer Of
 69:             0: (Adr: Long_Integer); 1: (Ptr: ^Integer);
                End;
 70:   Begin {$P-} X.Adr:=Adr; PeekWord:=X.Ptr^; {$P=} End;
 71:    Procedure PokeWord(Adr:Long_Integer; Wert:Integer);
 72:    Var X: Record Case Integer Of
 73:             0: (Adr: Long_Integer); 1: (Ptr: ^Integer);
                End;
 74:   Begin {$P-} X.Adr:=Adr; X.Ptr^:=Wert; {$P=} End;
 75:   Begin
 76:     Adr:=Malloc(Flang);
 77:     If Adr<=0 then InstallFont:=NIL else begin
 78:      Fread(FHandle,Flang,Adr);
 79:      Ptr:=FontPointer(Adr);
 80:     {$P-}
 81:       Anz:=SwapWord(Int(Shr(Ptr^.FontData,16)));
 82:      Ptr^.FontId:=SwapWord(Ptr^.FontId);
             Ptr^.FontSize:=SwapWord(Ptr^.FontSize);
 83:     For I:=18 to (Anz-1) div 2 do
 84:         PokeWord(Adr+I*2,SwapWord(PeekWord(Adr+I*2)));
 85:      Ptr^.HorOffsetTable:=Adr+Int(Shr
            (Ptr^.HorOffsetTable,16));
 86:      Ptr^.CharOffsetTable:=Adr+Int(Shr
            (Ptr^.CharOffsetTable,16));
 87:       Ptr^.FontData:=Adr+Int(Shr(Ptr^.FontData,16));
 88:      PrPtr:=LastFontPtr; PrPtr^.NextFont:=Ptr;
 89:      InstallFont:=Ptr; Ptr^.FontId:=FaceNo;
           Ptr^.NextFont:=NIL;
 90:     {$P=}
 91:    End;
 92:  End {of InstallFont};
 93:   Function LoadFont(Name:String;
         FaceNo:Integer): FontPtr;
 94:   Var Dat       : Text;
 95:       FHandle   : Integer;
 96:       Flang,I   : Long_Integer;
 97:       IOStat    : Boolean;
 98:    Function Fseek(Offs:Long_Integer; H,M:Integer):
          Long_Integer; Gemdos($42);
 99:   Function IO_State: Boolean; External;
100:   Begin
101:     IOStat:=IO_State; IO_Check(False);
102:     Reset(Dat,Name);
103:     If IO_Result<>0 then LoadFont:=NIL else begin
104:      FHandle:=Handle(Dat);
105:       Flang:=Fseek(0,FHandle,2); I:=Fseek(0,FHandle,0);
106:       LoadFont:=InstallFont(FaceNo,FHandle,Flang);
107:    End;
108:     IO_Check(IOStat);
109:  End {of LoadFont};
110:   Function LoadZsaFonts(Name:String; FaceNo:Integer;
         Var List:FontList): Integer;
111:   Var Dat         : Text;
112:       FHandle,I : Integer;
113:       Lang        : Long_Integer;
114:       IOStat,
115:       Fehler      : Boolean;
116:       Test        : String[8];
117:       TChar       : Char;
118:       FaceAnz,Fa,
119:       GrAnz,Gr    : Integer;
120:       Ptr         : FontPtr;
121:       LPtr        : FontList;
122:   Function IO_State: Boolean; External;
123:    Function Fseek(Offs:Long_Integer; H,M:Integer):
          Long_Integer; Gemdos($42);
124:   Function ReadInt: Long_Integer;
125:    Type IntegerPtr = ^Integer;
126:    Var I: IntegerPtr;
127:     Procedure Fread(H:Integer; C:Long_Integer;
           C,Buf:IntegerPtr); Gemdos($3F);
128:    Begin
129:      I:=NIL; New(I);
130:      FRead(FHandle,2,I);
131:      ReadInt:=(I^&$FFFF);
132:     Dispose(I);
133:   End;
134:   Begin
135:     IOStat:=IO_State; IO_Check(False);
136:     Reset(Dat,Name);
137:     If IO_Result<>0 then LoadZsaFonts:=0 else begin
138:      Test:='';
139:      For I:=1 to 4 do Read(Dat,TChar);
140:      For I:=1 to 8 do begin Read(Dat,TChar);
             Test[I]:=TChar; End;
141:      Test[0]:=Chr(8);
142:      If Test='GEM-FONT' then begin
143:        FHandle:=Handle(Dat); Lang:=FSeek(12,FHandle,0);
144:       FaceAnz:=ReadInt;
145:       Fehler:=False;
146:        For Fa:=FaceNo To FaceNo+FaceAnz-1 do begin
147:         GrAnz:=ReadInt;
148:         For Gr:=1 to GrAnz do begin
149:          If not Fehler then begin
150:           Lang:=ReadInt;
151:            Ptr:=InstallFont(Fa,FHandle,Lang);
152:           If Ptr=NIL then begin
153:             Fehler:=True; FaceAnz:=Fa-FaceNo; end
154:           else begin
155:            New(LPtr);
156:             LPtr^.Pointer:=Ptr;
157:             LPtr^.Next:=List;
158:            List:=LPtr;
159:          End;
160:         End;
161:        End;
162:      End;
163:       LoadZsaFonts:=FaceAnz;
164:      End else LoadZsaFonts:=0;
165:    End;
166:     IO_Check(IOStat);
167:  End {of LoadZsaFonts};
168:   Procedure UnloadFont(Ptr:FontPtr);
169:   Var P: FontPtr;
170:    Procedure FMfree(Adr:FontPtr); Gemdos($49);
171:   Begin
172:     {$P-} P:=FirstFontPtr;
            While P^.NextFont<>Ptr do P:=P^.NextFont;
173:     P^.NextFont:=Ptr^.NextFont; {$P=}
174:    FMfree(Ptr);
175:  End {of UnloadFont};
176:   Procedure UnloadFonts(Var List:FontList);
177:   Var P: FontList;
178:   Begin
179:     P:=List;
180:     While P<>NIL do begin
181:        UnloadFont(P^.Pointer);
182:       List:=P;
183:       P:=P^.Next;
184:       Dispose(List);
185:    End;
186:     List:=NIL;
187:  End {of UnloadFonts};
188:   Procedure Text_Face(FaceNo:Integer);
189:   Var Control : Ctrl_Parms;
190:       Int_in  : Int_In_Parms; Int_out : Int_Out_Parms;
191:       Pts_in  : Pts_In_Parms; Pts_out : Pts_Out_Parms;
192:   Begin
193:     Int_in[0]:=FaceNo;
194:       VDI_Call(21,0,1,0,Control,Int_in,Int_out,Pts_in,
            Pts_out,False);
195:  End {of Text_Face};
196:
197:  { END OF FONTSUBS.PAS }
```



STARKE SOFTWARE

ST ARCHIVAR
DIE ÜBERZEUGENDE DATEIVERWALTUNG

- Unterstützt die Produktion wissenschaftlicher Texte
- Verwaltet und bearbeitet Zitate u. Literaturangaben und Ausgabe als Text
- Komfortable Nutzung für Video- u. Adressendateien
- Datenverwaltung mit Dateien von bis zu 19 Datenfeldern (38 in Vorbereitung und 200 Datensätzen
- Dynamisch erweiterbar
- Veränderung der Dateimasken
- Speicherung der Daten als DIF oder SDF Datei, damit Übernahme in Datenbank möglich
- Beliebige Programme nachladbar u. auszuführen ohne ARCHIVAR zu verlassen (z. B. 1st Word)
- mit allen Textprogrammen kompatibel, die ASCII einlesen
- Ausdruck von Karten (z. B. Bibliothekskarten)
- Listen-Ausdruck auf Knopfdruck
- Etiketten-Ausdruck
- Einfache Druckerprogrammierung
- Einfach bedienbar mit der Maus

ST-ARCHIVAR enthält zahlreiche Hilfsprogramme

- Bibliotheks-Suchfunktion (ganze Disketten durchsuchen lassen)
- Uhr-Einstellung
- Info-Datei über alle ARCHIVAR-Dateien erstellen
- Druckersteuer-Programm
- Kopierprogramm
- Sortierprogramm
- Erzeugt RAM-DISK G bis 1750 KB
- Automatisches Kopieren in die RAM DISK G beim Start

und das alles für **DM 89,-**

ST PRINT
4 NÜTZLICHE PROGRAMME IN EINEM

- **RESETFESTE RAMDISK**
- Größe einstellbar von 32 – 4000 KB
- kann auf Laufwerk C bis P gelegt werden
- arbeitet problemlos mit einer Harddisk
- sehr schnell, da in Maschinensprache programmiert
- **DRUCKERSPOOLER,**
- Größe einstellbar von 2 – 510 KB
- Arbeitet mit TOS- u. GEM-Programmen
- Hohe Geschwindigkeit beim Ausdruck
- **DRUCKERVOREINSTELLUNG**
- mit der Maus, Knopfdruck statt Handbuch
- viele Einstellmöglichkeiten
- Einstellung des Druckers vom Desktop u. aus jedem GEM-Programm (VIP Prof., Wordplus, Tempus etc.) möglich.
- **HARDCOPYROUTINE**
- nutzt die Fähigkeiten von 9-, 18- u. 24-Nadeldruckern
- versch. Auflösungen, Schnelldruck bis Qualitätsdruck
- Umsetzung der Farben in Grauwerte
- gespoolte Hardcopy etc.

DM 59,-

ST PLOT
KURVENDISKUSSIONS- U. FUNKTIONSPLOTTER-PRG.

- ist ein Programm, das mehr kann, als einfach nur Funktionen zeichnen. Es bestimmt die richtigen Ableitungen u. damit werden Null-, Extrem- und Wendestellen berechnet, es stellt den Definitionsbereich u. die Periode fest und vieles andere mehr.
- zeichnet die Schaubilder von bis zu 3 Funktionen u. deren erste u. zweite Ableitung gleichzeitig auf einen Bildschirm. Danach können Ausschnitte der Funktionen vergrößert und verkleinert werden.
- ist komplett GEM- und mausgesteuert. Die Bedienung ist somit sehr einfach. Sie brauchen nur die Funktion einzugeben, alle anderen Berechnungen erledigt für Sie das Programm.
- läuft sowohl in mittlerer wie auch in hoher Auflösung.
- ist nicht nur für Professoren, Studenten u. Schüler interessant. Es kann überall dort angewendet werden, wo man sich mit Funktionen beschäftigt.
- Umfangreiche Eingabemöglichkeiten, z. B. Verwendung sämtl. Funktionen eines wiss. Rechners. Defin. von 10 versch. Zahlenwerten als Konstanten, e u. Pi verwendbar.
- Komfortable Zeichnung, z. B. autom. Zeichnung u. Beschriftung der Achsen und deren Skalierung, Fehlerroutinen, damit keine unnötigen Asymptoten gezeichnet werden, beliebig viele Funktionen nacheinander in ein Bild einzeichenbar ermöglicht Funktionenvergleich u. Ablesung der Schnittpunkte etc.
- Beste Berechnung der Funktionswerte, z. B. Hohe Rechengeschwindigkeit, hohe Rechengenauigkeit.

DM 59,-

ST AKTIE
AKTIENVERWALTUNG

- Aktien- u. Depotverwaltung u. Bilanzierung
- CHART-Analyse incl. Datenbank mit Kurven mit über 50 versch. Aktien beliebig aktualisierbar u. erweiterbar
- Sämtl. großen deutschen Standardwerte (55 St.) bis Okt. 87 fortgeführt
- Kto.-Führung für alle Orders u. andere Buchungen
- Kompakt, übersichtlich, Menü-orientiert, einfache Bedienung durch Maus, u. lfd. Programmhilfen, Graph. Darstellung standardgemäß, generiert selbständig

DM 69,-

ST DIGITAL
LOGIKSIMULATOR FÜR DEN ATARI ST

Ein Programm zum Erstellen, Testen und Analysieren von Logikschaltungen für Ausbildung und Hobby-Elektronik.

- Komfortable GEM-Umgebung
- Bauteile lassen sich per Maus plazieren u. verdrahten
- Umfassendes Bauteile-Set (Grundgatter, Ein-/Ausgabe-Bausteine, Flip-Flops, etc.)
- Definition zusätzlicher Bauteile durch Makrotechnik
- Makros können in Libraries gespeichert werden
- Interaktive Simulation mit Darstellung der Leitungszustände (d. h. Betätigung von 'Schaltern' mit der Maus und sofortige Reaktion der Schaltung)
- Erzeugen von Impulsdiagrammen
- Hardcopy-Funktion
- Hardwarevoraussetzung: ST/Monochrom-Monitor

DM 89,-

SKYPLOT plus
EIN ASTRONOMIEPROGRAMM FÜR STERNENFREUNDE

- zwei Sterndatensätze mit 612 bzw. 15 383 Sternen
- 1053 Nebel, Sternhaufen u. Galaxien
- acht Planeten, Sonne, Mond u. drei Kometen vordefiniert
- Eingabe eigener Bahnelemente möglich
- Darstellung des Himmels als rechteckige Karte
- maßstäbliche Darstellung von Nebeln u. Planeten als Scheiben, letztere als Phrasen – etc. dadurch Darstellung von Finsternissen, Durchgängen, Bedeckungen etc.
- automat. Suche nach Konjunktionen
- Bewegungsbahnen von Planeten mit max. 32 000 Punkten, speicherabh.
- einstellbarer Datumsbereich 1.1.1583 – 31.12.3000
- Gradnetz auf Wunsch einblendbar
- für Hardcopy Bild invertierbar
- Darstellung dreidimensional oder von Planeten anderer Sterne aus
- Stellarstatistik; Sichtbarkeitsdiagramme
- Sternkarten können mit allen Objekten auf Plotter ausgegeben werden
- Simulation eines Films im Speicher, dadurch Bewegungsabläufe sichtbar
- Handbuchfunktion verweist bei jedem Menüpunkt auf Handbuchseite
- und vieles andere mehr
- Hardware-Voraussetzung: ATARI ST mit mind. 1 MByte.

DM 198,-
Update DM 80,-

ST-LEARN
DER VOKABELTRAINER FÜR DEN ST

programmiert v. D. Owerfeldt, Gewinner der GOLDENEN DISKETTE '87 für das beste Lernprogramm.

- Läuft vollständig unter GEM
- „Intelligente" Auswertung der Benutzereingaben
- Fehlerhäufigkeit einer Vokabel wird berücksichtigt
- Berücksichtigung mehrerer Bedeutungen eines Wortes
- Vielfältige Möglichkeiten des Lernens und der Abfrage
- Integriertes Lernspiel „HANGMAN"
- Spezielle Auswertung für unregelmäßige Verben (bei Eingabe von „to go" werden die anderen 2 Formen nachgefragt)
- Bei offensichtlicher Ähnlichkeit der Wörter wird wahlweise ein 2. Versuch zugelassen
- Trotz Einordnung der Vokabeln nach Lektionen oder Wissensgebieten ständig schneller Zugriff auf **alle** Vokabeln (nur durch Größe des Speichermediums begrenzt)
- Voller europäischer Zeichensatz (Zugriff durch die Maus unter GEM)
- Auch für Farbmonitor in mittlerer Auflösung
- Wörterbuchfunktion durchsucht alle Files eines Speichermediums nach einer Übersetzung ab
- Verbessertes Eingabeformular

DM 59,-

ST STRUKTUR-PAINTER
DAS SUPER-PROGRAMM zum kreativen Erstellen von Graphiken aller Art – ohne Malkenntnisse

- Einfache Bedienung, ansprechende Menüs
- Automatischer Malmodus, Muster in Millionen Variationen erstellt
- 2 Füllroutinen, 2 Blockspeicher
- WYSIWYG
- Ausschneiden u. Einkopieren von beliebigen Formen
- Freihand, Verschiedene Pinsel, Füllmuster als Pinsel, Malen mit Bildausschnitten
- Füllmuster definieren ohne Editor; eine ganz neue Methode
- Füllmuster-Bibliotheken; über 200 Füllmuster gleichzeitig verfügbar
- Vergrößern, Verkleinern; 3 Endprodukte zur Auswahl
- Spiegeln, Drehen (dreht in 2 Richt. gleichzeit.)
- Stauchen
- Rahmen (1): Erzeugen von Rahmen aus allen Graphiken
- Rahmen (2): Automatische Rahmenberechnung
- Schatten: Automatische Schattenberechnung (3-D-Effekt)
- Teilmuster; Font's (23 Größen, 21 Arten, 4 Verknüpfungen
- Lineal: Einblendbare Einteilung
- Radierer u. Sprühdose beliebig einstellbar
- Druckertreiber für alle Epson- u. IBM-kompat. Drucker sowie Laserdrucker implementiert
- Schnelle Lupe mit: Punkt, Rechteck, Linie, Invertieren, Löschen
- Invertieren auf Knopfdruck während gezeichnet wird
- Voll Mausgesteuert
- Keine Kompatibilitätsprobl. mit Bildern anderer Programme
- STruktur-Painter Bilder auch mit Signum 2 verarbeitbar.
- und viele weitere Möglichkeiten
- Hardwarevoraussetzung: ST mit mind. 512 KB frei

Ein Spitzenprogramm **DM 89,-**

* alle Preise sind unverbindlich empfohlene Verkaufspreise

Bitte besuchen Sie uns in Halle 7 / Stand E 46



16. – 23. MÄRZ 1988

Heim Verlag

Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt
Telefon 0 61 51 - 5 60 57

BESTELL-COUPON

an Heim-Verlag
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt

Bitte senden Sie mir: ______________________________

____

zzgl. DM 5,- Versandkosten (unabhängig von bestellter Stückzahl)
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei

Name, Vorname ______________________________

Straße, Hausnr. ______________ PLZ, Ort ______________

Benutzen Sie auch die in ST COMPUTER vorhandene Bestellkarte.

```
 1:  ; Listing 2b:
 2:  ; ===========
 3:  ;
 4:  Start
 5:   movem.l d0/a0-a2,-(a7)      ; Register retten
 6:   dc.w $a000                  ; Line-A-Init
 7:   movea.l 4(a1),a1            ; Adresse des ersten
 8:   movea.l 84(a1),a1           ; GEM-Fonts bestimmen
 9:   lea Reg(Pc),a0              ; dort soll diese Adresse
                                    hin...
10:   move.l a1,(a0)              ; also machen wir das
11:   movem.l (a7)+,d0/a0-a2      ; Register restaurieren
12:   rts                         ; und zurück zum Pascal...
13:   nop
14:  Reg ds.l 1
15:  End
```

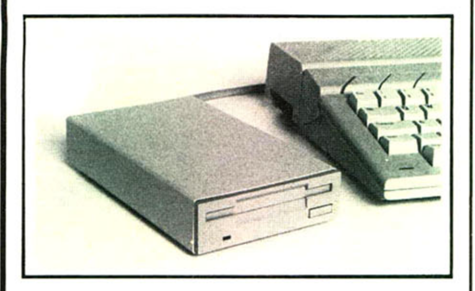
```
 1:  { Listing 3: }
 2:  { ========== }
 3:
 4:  {$U100}
 5:  Program Font_Test;    { 1987 by LPSoft Lutz Preßler
        24.11.1987 }
 6:
 7:  Const    {$IGEMCONST}
 8:
 9:  Type     {$IGEMTYPE}
10:           {$IFONTTYPE}
11:
12:  Var      Zeichensatz                : Font;
13:           Pfad,ZeichsName,FaceNa     : String;
14:           Thoch,Attribute            : Integer;
15:           Ok,Zsa                     : Boolean;
16:           Al,Anzahl,Face,I           : Integer;
17:           ZListe                     : FontList;
18:
19:  {$IGEMSUBS}
20:  {$IFONTSUBS}
21:
22:  Procedure Taste; Gemdos(7);
23:
24:  { Procedure Text_Heigth(Height:Integer); External;
25:    bei mir in GEMSUBS.PAS deklariert.                 }
26:   Procedure DefText(Color,Style,Rotation,Height,
        Face:Integer);
27:   Begin
28:     If Face>0 then Text_Face(Face);
29:     If Color>=0 then Text_Color(Color);
30:     If Style>=0 then Text_Style(Style);
31:     If Rotation>=0 then Text_Rotation(Rotation);
32:     If Height>0 then Text_Heigth(Height);
33:  End;
34:
35:   Procedure FaceName(FaceId:Integer;
     Var FaceName:String);
36:   Var Control  : Ctrl_Parms;
37:       Int_in   : Int_In_Parms; Int_out  : Int_Out_Parms;
38:       Pts_in   : Pts_In_Parms; Pts_out  : Pts_Out_Parms;
39:       I        : Integer;
40:   Begin
41:     Int_in[0]:=FaceId;
42:       VDI_Call(130,0,1,0,Control,Int_in,Int_out,Pts_in,
          Pts_out,False);
43:     I:=1;
44:     Repeat
45:        FaceName[I]:=Chr(Int_out[I]); I:=I+1;
46:     Until (I=35) or (FaceName[I-1]=Chr(0));
47:     FaceName[0]:=Chr(I-2);
48:  End;
49:
50:  Procedure TestAusgabe(F:Integer);
51:  Begin
52:     Write('Texthöhe(Pixel)  : '); ReadLn(Thoch);
53:     If IO_Result<>0 then Thoch:=13;
54:     Write('Textattribute     : '); ReadLn(Attribute);
55:     If IO_Result<>0 then Attribute:=0;
56:      DefText(-1,Attribute,-1,Thoch,F);
57:    Hide_mouse;
58:     Draw_String(10,200,'Dies ist ein
          GEM-Zeichensatz-Test. Lutz Pre_ler');
59:    Show_mouse;
60:     DefText(-1,0,-1,13,1);
61:  End;
62:  Procedure Bildschirm_loeschen;
63:  Begin Hide_mouse; Clear_Screen; Write(Chr(27),'H');
        Show_mouse; End;
64:
65:  Begin If Init_Gem>=0 then begin
66:     Bildschirm_loeschen;
67:     Set_Clip(0,0,640,400);
68:     WriteLn('Letzter Font HEX: ',
             FontAdress(LastFontPtr):8:h);
69:   Repeat
70:       Al:=Do_Alert('[1][LPSoft-Pascal-GEM-Font-Demo|
          Welche Zeichensatz-Art?][FNT|ZSA|ENDE]',1);
71:     Ok:=(Al<>3);
72:     If Ok then begin
73:      Zsa:=(Al=2);
74:       If Zsa then Pfad:='\*.ZSA' else Pfad:='\*.FNT';
75:      ZeichsName:='';
76:       Ok:=Get_In_File(Pfad,ZeichsName);
77:      If Ok then begin
78:       Bildschirm_loeschen;
79:       If Zsa then begin
80:         ZListe:=NIL;
81:          Anzahl:=LoadZsaFonts(ZeichsName,2,ZListe);
82:         If Anzahl=0 then WriteLn('FEHLER!') else begin
83:          WriteLn('Geladene Faces:');
84:          For I:=2 to Anzahl+1 do begin
85:            FaceName(I,FaceNa);
86:            WriteLn(I:2,': ',FaceNa);
87:          End;
88:          Taste; Bildschirm_loeschen;
89:          WriteLn('Letzter Font HEX: ',
                FontAdress(LastFontPtr):8:h);
90:          Repeat
91:            IO_Check(False);
92:            Write('Face (von 1 bis ',Anzahl+1,',
                  Ende=0): '); ReadLn(Face);
93:            If IO_Result<>0 then Face:=0;
94:           If Face>0 then begin
95:             TestAusgabe(Face);
96:             IO_Check(True);
97:             Taste;
98:           End;
99:            Bildschirm_loeschen;
100:         Until Face=0;
101:          UnloadFonts(ZListe);
102:          WriteLn('Letzter Font HEX: ',
                FontAdress(LastFontPtr):8:h);
103:       End
104:     End else begin
105:         Zeichensatz:=LoadFont(ZeichsName,2);
106:        If Zeichensatz=NIL then WriteLn('FEHLER!')
             else begin
107:         FaceName(2,FaceNa);
108:         WriteLn('Face-Name:         ',FaceNa);
109:         WriteLn('Letzter Font HEX: ',
                FontAdress(LastFontPtr):8:h);
110:         IO_Check(False);
111:         TestAusgabe(2);
112:         IO_Check(True);
113:         UnloadFont(Zeichensatz);
114:         WriteLn('Letzter Font HEX: ',
                FontAdress(LastFontPtr):8:h);
115:        Taste;
116:       End;
117:      End;
118:     End;
119:    End;
120:   Until Ok=False;
121:  Exit_Gem; End; End.
```

```
 1:  ' Listing 4:
 2:  ' ==========
 3:  '
 4:  ' Mehrere GEM (*.FNT) - Fonts zu einer Datei (*.ZSA)
       zusammenfassen.
 5:  ' 1987 LPSoft, Lutz Preßler
 6:  '
 7:  Print " GEM-Fonts zu einer *.ZSA-Datei zusammenfassen."
 8:  Print " ";String$(46,"-")
 9:  Print " Erzeuge Datei ...              ";
10:  Fileselect "\*.ZSA","",Ausgabef$
11:  Print Ausgabef$
12:  Open "O",#1,Ausgabef$
13:  Print #1,"ZSA-GEM-FONT";
14:  Input " Wieviele Faces ?               ",Faces%
15:  Print #1,Mki$(Faces%);
16:  For F%=1 To Faces%
```

geht weiter..

720 KB 3.5 Zoll
NEC FD 1037a
Laufwerk anschlußfertig 279,-

Superflaches Diskettenlaufwerk (25.4 mm hoch) elegantes, sehr kratzfestes Stahlblechgehäuse
Wichtig: Computertyp angeben!



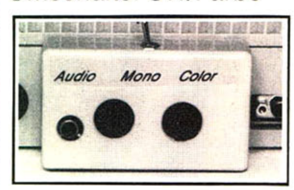
Floppygehäuse Stahlbl. Atari grau **29,-**
Für die Laufwerkstypen NEC 1036a, NEC 1037a, Fd 55 FR., kompl. mit Zugentlastung und Fußchen

5.25 Laufwerk 720 KB
anschlußfertig m. 40/80 Tr. Umsch. **359,-**
beige Frontblende ohne Aufpreis
Verwendung von sehr hochwertigen TEAC FD 55 FR Laufwerken

Einbau 2. Floppybuchse + 27,90
Umschaltung Drive A/B + 20,-
nur in Verbindung mit 2. Floppybuchse erhältlich. Ermöglicht das wahlweise Booten von Laufwerk A oder Laufwerk B

NEC FD 1037a 197,-
Teac FD 55 FR 259,- modif. 269,-

Switchbox ab 29,90
Umschalter SW/Farbe



Switchbox mit Monitorkabel 50 cm lang, Audioausgang o. Abb.
Bausatz **29,90**
Fertiggerät f. Mega ST **39,90**
Switchbox direkt an den Rechner ansteckbar (daher kein Kabelsalat mehr) siehe Abb. **49,90**

Floppyswitchbox 89,-
Ermöglicht den Anschluß von 3 Laufwerken am ST
Das Umschalten erfolgt ohne Reset. Die Switchbox ist mit Treiber und Original Buchsen ausgestattet.

Druckerswitchbox 1 Drucker an 2 Computer oder 2 Computer an einen Drucker	**89,-**
Tastaturgehäuse Lieferung komplett mit Spiralkabel Resettaster und Joystickbuchsen eingebaut	**124,-**
Blitter TOS o. Rom TOS mit oder ohne Fast Load	**109,-**
Scartkabel Länge 2 m	**39,90**
Druckerkabel Länge 2 m	**29,90**
Fast ROM o. Blittertos -Fastrom o. Fast Load Ch. Hertz	**24,90**
Speichererweiterung steckbar (CSF) oder lötbar	**219,-**
Uhrmodul (extern)	**89,-**
Eprom-Bank 128 KB	**49,-**

Stecker und Buchsen, Kabel

Monitorstecker	6,40	Monitorbuchse	8,90
Floppystecker	8,90	Floppybuchse	10,90
Floppykabel Atari — 3.5 Zoll (Shugart)			27,90
Audiokabel Switchbox ST ca. 2.5 m			7,90

Neuheiten

Automon **74,-**
Softwaremäßige Umschaltung zwischen SW- u. Farbmonitor. Umschaltung erfolgt ohne Reset.

AT Tastatur am ST **189,-/289,-**
Adapterbox zum Anschluß einer AT Tastatur am ST
Einbautyp/Fertiggerät

ST Software

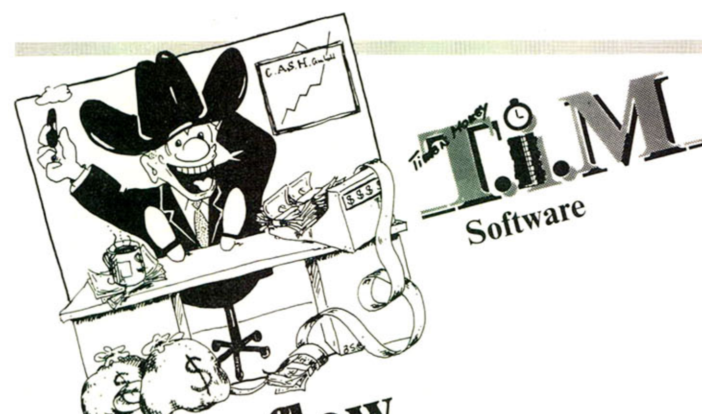
Signum II	Preis auf Anfrage
STAD	Preis auf Anfrage
Omikron. Basic	159,-
Omikron. Comp.	159,-
PC Ditto	197,-

Public-domain (alle ST Disketten)
incl. Diskette eins. format. DM 6,40
Liste gegen DM 2,- in Briefmarken

1 Paket — 5 Disk. doppels. format. nur 35,-

P.1 Die besten Spielprogr. z. B. Solitaire, Metropol, Mac Pan .....
P.2 Die besten Utilitieprogr. z. B. Fcopy, Diskchecker, Ramdisk .....
P.3 Die besten Mal- u. Soundp. z. B. E-Plan, Soundcomputer, Andromeda .....
P.4 Die besten diversen Prg. z. B. Etikettdat., Aktien, ST-Calc .....

Hard- u. Soft A. Herberg, Bahnhofstr. 289, 4620 Castrop-Rauxel, Tel. (02305) 74258 —Händleranfragen erwünscht—



Cashflow
Ein Kassenbuch

Die zeitgemäße Form des Kassenbuchs
→ sofort: Kassenbestand
→ sofort: Negativbestand
→ sofort: Informationen
anfordern bei uns oder Ihrem Fachhändler

Händlerverzeichnis siehe Seite 66

Unverb. Preisempfehlung:
Cashflow DM 298,–;
Handbuch DM 30,–*;
Demo 10,–;
*wird bei Direktkauf angerechnet.

C.A.$.H. GmbH
Robert-Bosch-Straße 20 a
8900 Augsburg
Telefon 08 21 / 70 38 56

DISKETTENLAUFWERKE

vollkompatibel, anschlußfertig, inkl. Kabel, Netzteil, Metallgehäuse, 2 x 80 Tracks, 1 MB unform., 3 ms Steprate, Test in 68000er 8/87

25,4 mm High Tech

3,5" Qualitätslaufwerke, 25,4 mm flach, modernste Technik, anschlußfertig und vollkompatibel. Netzteile mit **VDE** und **SEV.**

NEC 1037 oder **TEAC FD 135** **298.-**

Diskettenlaufwerke:

NEC FD 1037	199.-
TEAC FD 55 FR	229.-
TEAC FD 135 FN	209.-

3,5" Doppelstation 549.-
2 MB (unformatiert) bestückt mit NEC 1037 oder TEAC FD 135

Millionenfach bewährt:



TEAC 5.25"

FD 55 FR, 1 MB, 2 x 80 Tracks, integr. Netzteil, anschlußfertig und vollkompatibel
incl. 40/80 Tracks
unterstützt MSDOS Emulatoren wie z. B. PCDitto **349,–**

Y-Adapter zum Anschluß von 2 FSE Laufwerken, Drive Select schaltbar **49.-**

Disketten: 3,5" MF 2 DD, 135 tpi ab 10 St. **2,50**, ab 100 St. **2,30**

Frank Strauß Elektronik
St. Marienplatz 7 6750 Kaiserslautern Tel. 0631/16258

```
17:     Print " Face ";F%;" : Wieviele Größen ? ";
18:     Input "",Groessen%
19:     Print #1,Mki$(Groessen%);
20:     For G%=1 To Groessen%
21:       Print " Grö_e ";G%;" ...                    ";
22:       Fileselect "\*.FNT","",Name$
23:       Print Name$
24:       Open "I",#2,Name$
25:       Laenge%=Lof(#2)
26:       Print #1,Mki$(Laenge%);
27:       While Laenge%>32000
28:         Print #1,Input$(32000,#2);
29:         Sub Laenge%,32000
30:       Wend
31:       Print #1,Input$(Laenge%,#2);
32:       Close #2
33:     Next G%
34:   Next F%
35:   Close #1
36:   Print " Ok."
37:   Void Inp(2)
38:   Edit
```

```
1:  ' Listing 4:
2:  ' ==========
3:  '
4:  ' *** Font-Konvertierung von STAD- oder
        Fontloader-Format zu GEM-Font ***
5:  ' *** 1987 by LPSoft Lutz Pre_ler, Sandkrug,
        Ahornweg 11, 2904 Hatten ***
6:  '
7:  Repeat
8:    Cls
9:    Alert 1,"Font-Konvertierung , LPSoft|STAD- oder
          Fontloader-Fonts|  in GEM-Fonts umwandeln.",1,
          "STAD|FONTLOAD|ENDE",Res
10:   If Res=3
11:     Edit
12:   Endif
13:   '
14:   Fileselect "\*.FNT","",A$
15:   If A$="" Or Right$(A$)="\"
16:     Edit
17:   Endif
18:   If Exist(A$)
19:     Open "I",#1,A$
20:     Chars%=(Lof(#1)/16*Res)+32*(Res=2)
21:     If Chars%>256 Or Chars%=0 Or (Res=2 And Chars%<65)
22:       Print "Falsches Datei-Format!"
23:       Void Inp(2)
24:       Edit
25:     Endif
26:     If Res=2 And Chars%>96
27:       Chars%=96
28:     Endif
29:     Dat$=Space$(Lof(#1)*Res+256*(Res=2))
30:     '
31:     If Res=1
32:       For I=0 To Chars%-1
33:         For B=0 To 15
34:           Poke Varptr(Dat$)+I+B*Chars%,Inp(#1)
35:         Next B
36:       Next I
37:       Lowade%=0
38:     Else
39:       For I=0 To Chars%+31
40:         If I>64 And I<=96
41:           For B=0 To 7
42:             Void Inp(#1)
43:           Next B
44:         Else
45:           J=I+32*(I>96)
46:           For B=0 To 7
47:             Poke Varptr(Dat$)+J+B*2*Chars%,Inp(#1)
48:             Poke Varptr(Dat$)+J+(B*2+1)*Chars%,
                  Peek(Varptr(Dat$)+J+B*2*Chars%)
49:           Next B
50:         Endif
51:       Next I
52:       Lowade%=32
53:     Endif
54:     '
55:     Highade%=Lowade%+Chars%-1
56:     If Res=2
57:       Name$="Fontloader-Font"
58:     Else
59:       Name$="STAD-Font"
60:     Endif
61:     Print "Fontname: ";
62:     Form Input 31 As Name$
63:       Name$=Left$(Name$+String$(32,0),32)
64:     '
65:       Head$=Mki$(0)+Mki$(&HA00)+Name$
66:       Head$=Head$+Chr$(Lowade%)+Chr$(0)
67:       Head$=Head$+Chr$(Highade%)+Chr$(0)
68:     Restore Header.data
69:     For I=1 To 18
70:       Read Za$
71:         Head$=Head$+Mki$(Val("&H"+Za$))
72:     Next I
73:       Head$=Head$+Chr$((88+2*(Chars%+1))
            Mod 256)+Chr$((88+2*(Chars%+1))\256)+Mki$(0)
74:       Head$=Head$+Chr$(Chars% Mod 256)+Chr$(Chars%\256)+
            Mki$(&H1000)+Mkl$(0)
75:     For I=0 To Chars%
76:         Head$=Head$+Chr$((I*8) Mod 256)+Chr$((I*8)\256)
77:     Next I
78:     '
79:     Header.data:
80:       Data 0D00,0B00,0800,0200,0200,0700,0800,0100,0700,
            0100,0100,5555,5555,0800,0000,0000,5800,0000
81:     '
82:     Fileselect "\*.FNT","",A$
83:     If A$<>"" And Right$(A$)<>"\"
84:       Open "O",#2,A$
85:         Bput #2,Varptr(Head$),88+2*(Chars%+1)
86:       Bput #2,Varptr(Dat$),Len(Dat$)
87:     Endif
88:     Close
89:     Print "Ok."
90:   Endif
91:     Alert 2,"Noch einen Font umwandeln?",1,"JA|NEIN",Res
92: Until Res=2
93: Edit
```

```
1:  ; *** FONTLOADER - STAD-Font als neuen Systemfont ***
2:  ; ***       (sowohl GEM als auch TOS) installieren.    ***
3:  ; *** auch für AUTO-Ordner(ohne Param.=SYSFONT.FNT)  ***
4:  ; *** 10.12.1987 by LPSoft,Lutz Pre_ler,2904 Hatten ***
5:  ;
6:  ; Konstantendefinitionen
7:  GemDos equ 1
8:  Keep equ $31
9:  FOpen equ $3D
10: FClose equ $3E
11: FRead equ $3F
12: FSeek equ $42
13: FSetDTA equ $1A
14: FSFirst equ $4E
15: Bios equ 13
16: Xbios equ 14
17: GetRez equ 4
18: SupExe equ 38
19: ;
20: ; Programmlänge berechnen
21:  move.l 4(a7),a0
22:  move.l #Text,d6
23:  sub.l a0,d6          ; d6 = zu reserv. Bereich
24: ;
25:  adda.l #129,a0
26:  movea.l a0,a6        ; Adresse der Commandline
27:  clr.l d0             ; Null-Byte ans Ende setzen
28:  move.b -1(a0),d0
29:  clr.b (a0,d0.b)
30: ; Auflösung überprüfen
31:  move #GetRez,-(a7)
32:  trap #XBios
33:  addq.l #2,-(a7)
34:  lea WrongRes,a0
35:  cmpi #2,d0
36:  bne Error    ; Fehlermeldung
37: ;
38:  move.l #Ok,Text      ; Default-Text nach Installation
39: ;
40: ; Programm schon resident?
41:  move.l #GetTrap,-(a7)
42:  move #SupExe,-(a7)
43:  trap #XBios
44:  addq.l #6,a7
45: ; movea.l OldTrap,a0
46:  clr.b d2
```

geht weiter...

```
47:    movea.l #FontData,a4    ; Adresse des Fontbereichs
48:    cmpi.l #"Lutz",-4(a0)
49:    bne.s NichtRes
50:    move.b #$ff,d2          ; Flag für "resident" setzen
51:    movea.l -12(a0),a4      ; jetzt vorhandene Fontadresse
52:    move.l -8(a0),OldFont   ; Adresse des Original-Fonts
53:    move.l -16(a0),GemDosTrap
54:    move.l -20(a0),OldTrap
55:   ; Commandline "#" oder "~"?
56:    move.b (a6),d0
57:    cmpi.b #"~",d0
58:    beq.s RAend
59:    move.b (a6),d0
60:    cmpi.b #"#",d0
61:    bne.s RWeiter
62:   RAend
63:    move.l #Sys,Text        ; ja: Ausgabetext ändern
64:   ;
65:    bra.s RWeiter
66:   NichtRes                 ; Programm noch nicht resident:
67:    movea.l #Info,a0        ; Info anzeigen
68:    bsr Message
69:    move.b (a6),d0          ; Commandline "~"
70:    cmpi.b #"~",d0
71:    beq.s RFehler           ; oder
72:    move.b (a6),d0
73:    cmpi.b #"#",d0          ; Commandline "#"
74:    bne.s RWeiter
75:   RFehler
76:    movea.l #Exit,a0        ; ja: geht nicht -> Hinweise
77:    bsr Message             ;     zur Anwendung anzeigen
78:    bsr Key                 ; Auf Tastendruck warten
79:    bra FResi               ; Programm normal beenden
80:   RWeiter
81:    move.b d2,Flag          ; "resident"-Flag abspeichern
82:   ;
83:    bsr FontLaden           ; ->Fontdatei(4096 Bytes)laden
84:   ; Font installieren
85:   ; Font-Adressen bestimmen
86:    dc.w $A000
87:    movea.l 4(a1),a3
88:    movea.l 84(a3),a3
89:    move.l a3,FontBase
90:    move.l 76(a3),d0
91:    tst.l d0
92:    beq FAuto
93:    move.l d0,OldFont
94:    move.l a4,76(a3)        ; Adresse eintragen
95:   FAuto
96:    move.l a4,FontAdr
97:   ; Font anmelden (direkt zugreifen)
98:    dc.w $A000
99:    move.l a4,-22(a0)
100:  ; Text ausgeben
101:   movea.l Text,a0
102:   bsr Message
103:  ; Commandline "~"?
104:   move.b (a6),d0
105:   cmpi.b #"~",d0
106:   bne.s TextEnde
107:   movea.l #NichtResiT,a0
108:   bsr Message
109:  TextEnde
110:   tst.b Flag              ; schon resident ?
111:   bne FResiX
112:  ; nicht resident, dann:
113:  ; Autoordner-Start?
114:   move.l #AutoTest,-(a7)
115:   move #SupExe,-(a7)
116:   trap #XBios
117:   addq.l #6,a7
118:   tst.b Auto
119:   beq Normal
120:   movea.l #AText,a0
121:   jsr Message
122:   move.l #SetDOS,-(a7)
123:   move #SupExe,-(a7)
124:   trap #XBios
125:   addq.l #6,a7
126:   bra.s ResEnd
127:  Normal
128:  ; Neuen GEM-Trap und GemDos installieren
129:   move.l #SetTraps,-(a7)
130:   move #SupExe,-(a7)
131:   trap #XBios
132:   addq.l #6,a7
133:   jsr Key
134:  ResEnd
135:  ; Programm resident lassen
136:   clr.w -(a7)
137:   move.l d6,-(a7)
138:   move.w #Keep,-(a7)
139:   trap #GemDos
140:  ;
141:  FResiX ; war schon resident:
142:   jsr Key
143:  ; Commandline "~"?
144:   move.b (a6),d0
145:   cmpi.b #"~",d0
146:   bne FResi
147:  ; Traps zurücksetzen
148:   move.l #ResetTraps,-(a7)
149:   move #SupExe,-(a7)
150:   trap #XBios
151:   addq.l #6,a7
152:  FResi  ; normal beenden
153:   clr -(a7)
154:   trap #GemDos
155:  ;
156:  SetDOS
157:   move.l 136,OldTrap
158:   bra.s SetDOS2
159:  SetTraps
160:   move.l 136,OldTrap
161:   move.l #GemTrap2,136
162:  SetDOS2
163:   move.l 132,GemDosTrap
164:   move.l #Trap1,132
165:   rts
166:  ResetTraps
167:   move.l OldTrap,136
168:   move.l GemDosTrap,132
169:   rts
170:  GetTrap
171:   move.l 136,a0
172:   rts
173:  AutoTest
174:   movem.l d0/d2/a0,-(a7)
175:   clr.b d2
176:   movea.l FontBase,a0
177:   move.l 76(a0),d0
178:   tst.l d0
179:   bne.s ExAuto
180:   move.b #$ff,d2
181:  ExAuto
182:   move.b d2,Auto
183:   movem.l (a7)+,d0/d2/a0
184:   rts
185:  ;
186:  FontLaden
187:  ; Commandline "#" oder "~"?
188:   move.b (a6),d0
189:   cmpi.b #"~",d0
190:   beq.s SysAdresse
191:   move.b (a6),d0
192:   cmpi.b #"#",d0
193:   bne.s Laden
194:  SysAdresse
195:  ; ja: Systemfontadresse in a4
196:  ; und Laden beenden
197:   movea.l OldFont,a4
198:   bra LEnde
199:  Laden
200:   move.b (a6),d0
201:   cmpi.b #"-",d0     ; - am Anfang
202:   bne.s DiskDa
203:  ; ja: Warten und Disk prüfen
204:   lea InsertDisk,a0
205:   bsr Message
206:   bsr Key
207:   addq.l #1,a6
208:   move.l #FontData,-(a7)
209:   move #FSetDTA,-(a7)
210:   trap #GemDos
211:   addq.l #6,a7
212:   move #31,-(a7)
213:   move.l #Dummy,-(a7)
214:   move #FSFirst,-(a7)
215:   trap #GemDos
216:   adda.l #8,a7
217:  ;
218:  DiskDa
219:  ; Commandline leer --> Default-Datei "SYSFONT.FNT"
```

geht weiter...



STARKE SOFTWARE FÜR STARKE COMPUTER

TKC-EINNAHME/ÜBERSCHUSS PLUS (Buchführung) **DM 149,–**
Automatische Führung von MWSt.-Konten. Ausgabe von Saldenliste, Bilanz, Journal, USt.-Voranmeldung. Bis zu 5 verschiedene MWSt.-Sätze, Abschlußzeitraum Monat, Quartal oder Jahr, universelle Druckeranpassung, ausführliches deutsches Handbuch!

TKC-HAUSHALT PLUS (Haushaltsbuchführung) **DM 129,–**
Überwachung aller Einnahmen und Ausgaben im Privathaushalt mit max. 250 Konten. Monats- und Jahresabschlüsse, Bilanzen und Journal, Saldenlisten mit Teilsummen für wählbare Kontengruppen, Daueraufträge mit beliebiger Frequenz, universelle Druckeranpassung, ausführliches deutsches Handbuch!

TKC-TERMIN/ADRESS (Terminplaner + Adressdatei) **DM 149,–**
Terminplaner mit Adressverwaltung, umfangreiche Suchfunktionen, Serienbriefe in Verbindung mit 1st Word, Terminfrequenz bei jedem Termin wählbar, Ausgabe von Termin- und Adresslisten, ausführliches deutsches Handbuch!

TKC-VIDEO (Videofilmverwaltung) **DM 79,–**
Verwaltet Ihre Videofilme unter GEM, umfangreiche Sortier- und Suchfunktionen, Listen- und Etikettendruck, Handbuch.

TKC-MUSICBOX (Musiktitelverwaltung) **DM 79,–**
Verwaltet bis zu 1500 LPs, CDs oder MCs, Titel-Suchfunktion, Listen- und Etikettendruck, Sortierung automatisch nach LP-Titeln! Druckeranpassung!

ST-MATHETRAINER (Trainingsprogramm für Grundschüler) **DM 59,–**
1x1, Grundrechnen, Umrechnung v. Gewichten und Längenmaßen, Benotung!

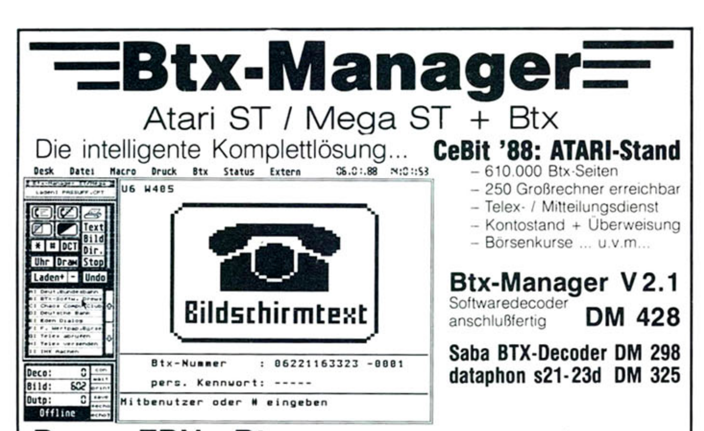
ST-RECHTSCHREIBEN (Trainingsprogramm für Grundschüler) **DM 59,–**
Singular & Plural, Interpunktion, u. a. Schwerpunktdatei für falsche Antworten!

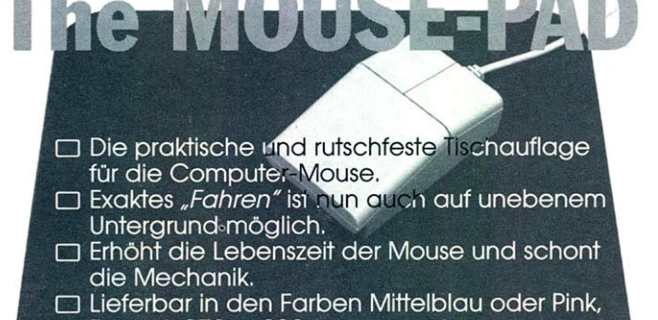
ST-KEYMASTER (Tastaturbelegungskit) **DM 49,–**
Tastaturbelegung in den Modi normal, shift und caps (nur S/W)!

ST-VOKABELTRAINER (Lernprogramm) **DM 49,–**
Sonderdatei für falsche Antworten, Druckerausgabe, %-Auswertung.

ST-GIRO (Utility für bargeldlosen Zahlungsverkehr) **DM 39,–**
Bedruckt Überweisungsträger, Formularinhalte speicherbar, ACC.

TK COMPUTER-TECHNIK THOMAS KASCHADT
BISCHOFSHEIMER STRASSE 17 · 6097 TREBUR-ASTHEIM · TELEFON (0 61 47) 5 50





MSM GmbH · Bayenthalgürtel 45 · 5000 Köln 51 · Tel. 02 21/38 20 20



```
220:    move.b -1(a6),d0
221:    tst.b d0
222:    bne.s NameOk
223:    move.l #Default,a6
224:  NameOk
225:  ; Datei öffnen
226:    move.w #0,-(a7)
227:    move.l a6,-(a7)
228:    move.w #FOpen,-(a7)
229:    trap #GemDos
230:    addq.l #8,a7
231:    move.l d0,d1
232:    lea NoFile,a0
233:    tst d0
234:    bmi Error
235:  ; Länge bestimmen
236:    move #2,-(a7)
237:    move d1,-(a7)
238:    clr.l -(a7)
239:    move #FSeek,-(a7)
240:    trap #GemDos
241:    adda.l #10,a7
242:    lea WrongSize,a0
243:    cmpi.l #4096,d0
244:    bne ClError
245:    move #0,-(a7)
246:    move d1,-(a7)
247:    clr.l -(a7)
248:    move #FSeek,-(a7)
249:    trap #GemDos
250:    adda.l #10,a7
251:  ;
252:  ; Datei etwas "verdreht" einladen
253:    clr.l d2
254:  Zeichen
255:    clr.w d3
256:  Bits
257:    clr.l d4
258:    movea.l a4,a5
259:    move.w d3,d4
260:    asl.w #8,d4
261:    add.l d2,d4
262:    adda.l d4,a5
263:    move.l a5,-(a7)
264:    move.l #1,-(a7)
265:    move.w d1,-(a7)
266:    move.w #FRead,-(a7)
267:    trap #GemDos
268:    adda.l #12,a7
269:    add.w #1,d3
270:    cmpi.w #16,d3
271:    bne.s Bits
272:    add.l #1,d2
273:    cmpi.l #256,d2
274:    bne.s Zeichen
275:  ; Datei schlie_en
276:    move.w d1,-(a7)
277:    move.w #FClose,-(a7)
278:    trap #GemDos
279:    addq.l #4,a7
280:  LEnde
281:    rts
282:  ;
283:  Trap1       ; neuer GemDos-Trap
284:    movem.l d0/d1/a0,-(a7)
285:    tst.b Auto
286:    beq TNormal
287:    move.l FontBase,a0
288:    move.l 76(a0),d0
289:    tst.l d0
290:    beq TrapOk
291:    bsr AutoInst
292:    bra TrapOk
293:  TNormal
294:    move.l 136,d0
295:    cmpi.l #$FC0000,d0
296:    bgt.s ChngTrap
297:    move.l $4f2,d1
298:    cmp.l d1,d0
299:    blt.s TrapOk
300:    move.l $4fa,d1
301:    cmp.l d1,d0
302:    bgt.s TrapOk
303:  ChngTrap
304:  ; GEM-Trap im TOS -> ändern
305:    move.l #GemTrap2,136
306:  TrapOk
307:    movem.l (a7)+,d0/d1/a0
308:    move.l GemdosTrap,a0
309:    jmp(a0)
310:  ;
311:  AutoInst
312:    move.l 136,OldTrap
313:    move.l #GemTrap2,136
314:    clr.b Auto
315:    movea.l FontBase,a0
316:    move.l 76(a0),OldFont
317:    move.l FontAdr,76(a0)         ; Adresse eintragen
318:    rts
319:  ;
320:  OldTrap
321:    dc.l 0
322:  GemDosTrap
323:    dc.l 0
324:  FontAdr
325:    dc.l 0
326:  OldFont
327:    dc.l 0
328:    dc.b "Lutz"
329:  GemTrap2    ; neuer GEM-Trap
330:    cmpi.b #$73,d0      ; VDI ?
331:    bne.s JumpGEM
332:    movem.l d0-d2/a0-a5,-(a7)
333:    movea.l d1,a4        ; a4 zeigt auf VDIPB
334:    movea.l (a4),a5      ; a5 zeigt auf Contrl-Feld
335:    cmpi.w #5,(a5)       ; Escape-Funktion ?
336:    bne.s ContGEM
337:    cmpi.w #102,10(a5)  ; Font anmelden ?
338:    bne.s ContGEM
339:    movea.l 4(a4),a5    ; a5 zeigt jetzt auf Intin-Array
340:  ; Font-Adressen bestimmen
341:    dc.w $A000
342:    move.l 8(a1),d2      ; -> TOS-Font
343:    movea.l 4(a1),a3
344:    movea.l 84(a3),a3   ; -> GEM-Font
345:    cmp.l d2,(a5)        ; TOS-Font anmelden ?
346:    bne.s ContGEM
347:    move.l a3,(a5)       ; ... dann GEM-Font anmelden.
348:  ContGEM
349:    movem.l (a7)+,d0-d2/a0-a5
350:  JumpGEM
351:    move.l OldTrap,a0
352:    jmp(a0)
353:  ;
354:  ; String ausgeben
355:  Message
356:    move.l a0,-(a7)
357:    move #9,-(a7)
358:    trap #GemDos
359:    addq.l #6,a7
360:    rts
361:  ; Auf Tastendruck warten
362:  Key
363:    move #7,-(a7)
364:    trap #GemDos
365:    addq.l #2,a7
366:    rts
367:  ; Fehlerbehandlung...
368:  ClError
369:    movea.l a0,a6
370:  ; Datei schlie_en
371:    move d1,-(a7)
372:    move #FClose,-(a7)
373:    trap #GemDos
374:    addq.l #4,a7
375:    movea.l a6,a0
376:  Error
377:    bsr.s Message
378:    bsr.s Key
379:  ; Programm normal beenden
380:    clr -(a7)
381:    trap #GemDos
382:  End
383:    .data
384:  Ok dc.b "Neuer Zeichensatz installiert.  (LPSoft,
        1987)",7,13,10,0
385:  Sys dc.b "Original Systemzeichensatz wieder
        angemeldet.",7,13,10,0
386:  NoFile dc.b "Fontdatei nicht vorhanden !!!",7,13,10,0
387:  WrongSize dc.b "Datei hat nicht das richtige
        Fontformat"
388:    dc.b " (falsche Länge) !!!",7,13,10,0
```

```
389:   WrongRes dc.b "Programm läuft nur in hoher
            Auflösung !!!",7,13,10,0
390:    InsertDisk dc.b "Bitte Diskette einlegen.",7,13,10,0
391:   Info dc.b "STAD-Font als neuen 8x16 GEM- und
            TOS-Systemzeichensatz"
392:     dc.b" installieren.",13,10
393:    dc.b "(c) 1987 by LPSoft, Lutz Pre_ler, Ahornweg 11,"
394:    dc.b" 2904 Hatten",13,10,0
395:   Exit dc.b 13,10,"ANWENDUNG:   FONTINST.TTP
             DATEINAM.FNT "
396:     dc.b "DATEINAM.FNT installieren.",13,10
397:    dc.b "    oder   FONTINST.TTP #    Originalzeichensatz"
398:     dc.b " wieder anmelden.",13,10,7,0
399:   NichtResiT dc.b "Programm nicht mehr resident.",13,10,0
400:   Dummy dc.b "\*.*",0
401:   Default dc.b "SYSFONT.FNT",0
402:   AText dc.b "Aus AUTO-Ordner gestartet.",13,10
403:    .even
404:    .bss
405:   ;
406:   FontBase ds.l 1
407:   TAuto ds.l 1
408:   Auto ds.w 1
409:   FontData ds.b 4096      ; Datenbereich für Font
410:   ; bis hier hin resident
411:   Text ds.l 1               ; Textadresse (Ok oder Sys)
412:   Flag ds.w 1               ; "resident"-Flag
413:   ;
414:   PrgEnd
```

Vom Papier zur Datenbank

1. Teil Datennormalisierung

Die Informationsverwaltung ist die klassischste aller Computeranwendungen. Eine Datenbank, sei es nun eine für Großrechner, in Form eines Telefonbuches oder eine für Personal-Computer, ist nichts anderes als eine sinnvoll organisierte Informationssammlung.

Für die Computer der ATARI ST-Serie sind mittlerweise eine Vielzahl von Datenbanken auf dem Markt. Sie sind alle hervorragend zum Verarbeiten von Massendaten geeignet, wie z.B. im kaufmännischen Bereich (Lagerverwaltung, Personaldaten, Buchführungen), wo bekanntlich Unmengen von verschiedenartigen Informationen anfallen.
Eine Grundregel: Müssen mehr als 50 Daten ständig im Zugriff sein, lohnt es sich, über den Einsatz eines Datenbanksystems nachzudenken. Der Hauptvorteil liegt im schnellen Zugriff auf einzelne Datensätze und im freien Sortieren nach bestimmten Feldern innerhalb eines Datensatzes.
Mein Kurs befaßt sich ausschließlich mit den relationalen Datenbanken, da ihre Bedeutung bereits heute beachtlich ist und in Zukunft noch steigen wird (Beispiel: PROLOG).
Um den Laien, wie aber auch allen anderen Interessierten, die Vorgehensweise beim Design einer Datenbank zu zeigen, ist dieser erste Teil des Kurses verfaßt. In weiteren geplanten Folgen werde ich ausgiebig auf die Datenmanipulationssprache sowie die Programmiersprache zum Erstellen lauffähiger Programme eingehen.
Ich werde heute den allgemeinen und prinzipiellen Aufbau relationaler Systeme erklären, die drei wichtigsten Regeln des Datenbankdesigns nennen und an anschaulichen Beispielen verdeutlichen.
Die folgenden Teile des Programmier-Kurses befassen sich dann mit speziellen Systemen, die auf dem ATARI verfügbar sind. Ich gehe besonders auf dBMAN Version 3.0 und STandard Base ein, weil sie sich am besten zum Schreiben professioneller Anwendungen eignen.
Außerdem bieten beide Programme eine Programmiersprache, die über die reine Datenmanipulation hinausgeht. Mit den zusätzlich implementierten Sprachelementen lassen sich leistungsfähige Anwendungssysteme schreiben, z.b. Buchführungen, Adreßverwaltungen oder Schallplattenarchive, die sehr einfach zu erstellen und zu erweitern sind und dennoch ein Höchstmaß an Benutzerfreundlichkeit aufweisen, obwohl die GEM-Oberfläche nicht genutzt wird.
Die Programmiersprache "erzwingt" regelrecht, strukturierte Programme zu schreiben (keine Sprunganweisungen, ein Befehl pro Zeile usw.).
Die mächtigen Befehle ermöglichen eine schnelle Programmierung von Plausibilitätskontrollen (um Falscheingaben zu unterbinden), so daß spezielle Tools (z.B. Resource Construction Set) nicht benötigt werden.
Ich meine auch, bei der Verarbeitung und Erfassung von Massendaten stört das GEM. Das ständige Wechseln von der Maus zur Tastatur verhindert einen "flüssigen" Arbeitsablauf.

Der grundsätzliche Aufbau relationaler Systeme

In einer Relation werden die Daten in Tabellenform gespeichert. Die Zeilen der Tabelle sind die Datensätze, die (mit Namen versehenen) Spalten der Tabelle sind die Felder. Es ergibt sich eine "flache" zweidimensionale Datei.
Zur einfachen und eindeutigen Identifikation eines Satzes (zum Wiederauffinden) dienen ein oder mehrere zusammengesetzte Felder, welche als Schlüssel (Key) deklariert werden, weil sie für jeden Satz einen bestimmten, sonst nicht vorkommenden Inhalt haben (Eindeutigkeit). Bei einem zusammengesetzten Key spricht man von einem Kombinationsschlüssel.
Mehrere Tabellen sind also über gemeinsame Datenfelder miteinander verbunden. Diese Beziehungen werden auch Relationen genannt, daher auch die Titulierung "relational". Die Gesamtheit aller Relationen bezeichnet man als Datenbank.

Die Charakteristik einer relationalen Datenbank:

Speicherorganisation für große Datenbestände mit einem System für deren effektive Wiederauffindung, bei der die Daten in sogenannten Relationen vorliegen. Auf die Tabellen können Mengenoperationen (Vereinigung, Durchschnitt, Differenz) sowie Relationenoperationen (Entfernung von Spalten, Auswahl von Zeilen, Verbinden von Tabellen)

```
Abbildung 1: Eine typische Relation
----------------------------------
  (Tabellenname)    (Schlüssel, Key)       (Feldname)
     TABELLE          Personal-Nr         Mitarbeiter
     ------------------=========-----------------------------
                   | 9900100 |           Egon Meier
                   | 9900107 |           Willi Würfel
                   | 9900114 |           Max Müller
      Tupel -->    | 9900121 |           Hans Hansen
                   \----+----/           -----+-----
                        |                     |
                      Domäne             Feldinhalt
```

Abbildung 1:Eine typische Relation

angewandt werden, mit deren Hilfe Datenauswahl und Datenmanipulation möglich sind. Der Zugriff auf die Daten erfolgt über die Werte der gespeicherten Daten, so daß für den Benutzer keine sichtbaren Zeiger und Verkettungen existieren.

Abbildung 1 zeigt eine typische Relation.

Jede Zeile mit gleichem Schlüssel wird als Tupel bezeichnet und ist in etwa mit einem Datensatz einer Datei vergleichbar (ein Tupel kann mehrere Sätze umfassen). Die Gesamtheit der verschiedenen Feldwerte eines Datenfeldes (das entspricht einer Spalte) heißt Domäne.

Das Schlüsselfeld sollte zur besseren Kenntlichkeit das erste Feld einer Tabelle sein (In meinen Beispielen ist der Key doppelt (======) unterstrichen).

DDL und DML

Für jedes Datenbanksystem gibt es eine spezielle Daten-Definitions-Sprache (Data Definition Language, DDL), mit deren Hilfe die Datenbank beim Anlegen in ihrem zukünftigen Aufbau beschrieben wird. Nachträgliche Änderungen (Hinzufügen von Feldern, Erweitern von Feldlängen usw.) sind problemlos möglich.

Mit der Data Manipulation Language (was heißt das wohl in deutsch?), DML, werden die Daten eingegeben, abgerufen und gepflegt.

Die Sprache der relationalen Systeme unterscheidet sich aber erheblich von denen, die die Daten hierarchisch oder nach dem Netzwerkmodell orientiert speichern.

Mit dem JOIN-Befehl (Vereinigung) kann aus zwei bestehenden Relationen eine neue, dritte Relation gebildet werden.

Unter Selektion versteht man das Ausblenden von bestimmten Zeilen einer Tabelle, mit Hilfe von Projektionen können Spalten aus der Sicht entfernt werden.

Mit diesen Befehlen verkleinert man die Datenbank auf die Spalten und Felder, die das jeweilige Anwendungsprogramm benötigt. Sie wird dadurch nicht physisch gelöscht; es erfolgt nur eine Eingrenzung der Betrachtung. Der vorherige Zustand ist jederzeit wiederherstellbar.

Diese Möglichkeiten zeigen den Vorteil relationaler Datenbanken: alle Operationen finden auf der Basis von Relationen statt. Das Ziel einer Operation ist dabei stets eine neue Relation, mit der weitergearbeitet werden kann.

Die Einfachheit der Datenmanipulationssprache ermöglicht endlich dem Endanwender den Zugang zu Datenbanken. Er braucht sich nicht wie ein Profi mit verschiedenen Satzarten, hierarchisch organisierten Zugriffspfaden und anderen schwer verständlichen Details zu befassen. Das Konzept einer Tabelle ist jedem zugänglich und hat im Grunde im mit elektronischer Datenverarbeitung nichts zu tun.

Der Anwender muß nur noch angeben, welche Daten er wünscht. Die Angabe, wie und wo die Daten zu finden sind, braucht er nicht zu machen. Dieses erledigt das Datenbanksystem, als Schnittstelle zwischen Mensch und Betriebssystem:

Anwendungprogramm <-> Datenbanksystem <-> Betriebssystem <-> DB (DB=Datenbank)

In der Realität kann der richtige Entwurf von Relationen durchaus ein schwieriger Prozeß sein, da es gilt, die Daten sinnvoll (auch im Hinblick auf die Zukunft) anzuordnen.

Eine große Weitsichtigkeit wird vom Programmierer verlangt, damit nicht ein unsauberes Design später zu katastrophalen Konsequenzen führt.

Die folgenden Schritte erläutern den richtigen Aufbau.

Die drei wichtigen Regeln des Datenbankentwurfs

Ich werde die drei Regeln an einem Beispiel beschreiben, da sie im eigentlichen Wortlaut nicht verständlich genug sind.

Ich wähle ein Beispiel aus der Praxis: das Verwalten von Disketten und Programmen.

In Abb. 2 zeige ich eine völlig unstrukturierte Tabelle, die in keiner Form normalisiert ist. Die Nachteile dieser "Relation" sind schnell aufgezählt:

a) es können nur maximal drei Programme pro Diskette gespeichert werden
b) die Suche nach einem bestimmten Programm dauert zu lange, da alle Felder durchforstet werden müssen
c) sind weniger als drei Programme auf einer Disk gespeichert, bleiben Felder unbelegt (Platzverschwendung)
d) Löschen einzelner Programme kann Schwierigkeiten bereiten

Abbildung 2: Beispiel einer nicht-normalisierten Relation

DISK	Disk-Nr	frei	belegt	Programm1	Programm2	Programm3
	1	100	700	GFA-Basic	GFA-Compiler	GFA Draft
	2	300	500	STAD	Campus	
	3	50	750	Signum!	Megamax C	

Abbildung 2: Beispiel einer nicht-normalisierten Relation

Um die Fehler zu beheben, entfernen wir die Mehrfachfelder (Programm1, Programm2, Programm3), deren Länge und Typ gleich sind, und schreiben sie untereinander weg. Damit führen wir die Relation in die 1. Normalform über (siehe Abbildung 3).

eine Datenredundanz (mehrfaches Speichern gleicher Daten) und daraus resultierend eine Verarbeitungsredundanz, da diese Information z.B. beim Löschen eines Programmes in mehreren Sätzen geändert werden muß.

DISK	Disk-Nr	frei	belegt	Programm
	1	100	700	GFA-Basic
	1	100	700	GFA-Compiler
	1	100	700	GFA-Draft
	2	300	500	STAD
	2	300	500	Campus
	3	50	750	Signum!
	3	50	750	Megamax C

Abbildung 3: Entfernen von Wiederholungsgruppen

DISK	Disk-Nr	frei	belegt
	1	100	700
	2	300	500
	3	50	750

PROGRAMM	Disk-Nr	Programm	Distributor
	1	GFA Basic	GFA Systemtechnik, Düsseldorf
	1	GFA Compiler	GFA Systemtechnik, Düsseldorf
	1	GFA Draft	GFA Systemtechnik, Düsseldorf
	2	STAD	Application System, Heidelberg
	2	Campus	digital Workshop, Bochum
	3	Signum!	Application System, Heidelberg
	3	Megamax C	Application System, Heidelberg

Abbildung 4: Auslagerung der Felder, die nur durch einen Teilschlüssel definiert werden

Die 1. Regel:

Alle Sätze eines Satztypes müssen dieselbe Anzahl von Feldern haben, d.h. Gruppen und multiple Felder sind verboten.

Demnach darf jedes Feld im Datensatz nur einmal und nicht mehrfach vorkommen.

Die Vorteile sind schnell erkannt: es können beliebig viele Programme pro Disketten-Nummer erfaßt werden. Alle vorher aufgezählten Nachteile scheinen beseitigt.

Leider sind nach der Umgestaltung einige Einträge mehrfach vorhanden. Die Informationen über die Diskettenkapazität werden mit jedem Programm erfaßt, was eigentlich garnicht notwendig wäre. Es entsteht

Man erkennt, daß die Felder FREI und BELEGT zwar von der Disketten-Nummer, aber nicht vom Programm-Namen, d.h. nur von einem Teilschlüssel, abhängig sind (der Key setzt sich hier wegen der Eindeutigkeit aus Disk-Nr. und Programmname zusammen).

Dazu werden alle Felder, die nur durch einen Teilschlüssel identifiziert werden, ausgelagert (in eine zweite Relation, siehe Abb. 4)

Die 2. Regel:

In Relationen mit einem Kombinationsschlüssel muß jedes nicht dazugehörige Feld vom gesamten Kombinationsschlüssel abhängen. Felder, die nur von einem Teil des Keys abhängen, werden mit diesem Feld als Schlüssel separat gespeichert (Entkopplung).

Die Datenbank liegt dann in der zweiten Normalform vor, wenn sie beide Regeln erfüllt (was hier gegeben ist).

Um noch die dritte Regel zu beschreiben, füge ich in die Tabelle PROGRAMM das Feld DISTRIBUTOR ein. Man erkennt schnell die entstehende Redundanz. Der Distributor wird mehrfach gespeichert. Damit auch die dritte Normalform erfüllt wird, entkoppeln wir nochmals in eine weitere Relation, DIST.

Zum Wiederauffinden vergeben wir eine interne Nummer. Sie wird als Fremdschlüssel bezeichnet, da sie nach außen hin nicht sichtbar wird.

Die 3. Regel:

Felder, die nicht Teil des Schlüssels sind, dürfen nicht untereinander abhängig sein. Ist das der Fall, werden auch diese voneinander abhängigen Felder in getrennten Relationen gespeichert.

Zum Erfüllen der dritten Normalform müssen alle drei Regeln eingehalten werden.

PROGRAMM	Disk-Nr	Programm	Distributor-Nr
	1	GFA-Basic	990091
	1	GFA-Compiler	990091
	1	GFA Draft	990091
	2	STAD	595555
	2	Campus	400000
	3	Signum!	595555
	3	Megamax C	595555

DIST	Distrubutor-Nr	Name	Anschrift
	400000	digital Workshop	Bochum, Kornhapener Str
	595555	Application System	Heidelberg, Brückenstr.
	990091	GFA Systemtechnik	Düsseldorf, Heerdter Str

Abbildung 5: Entkopplung aller Felder, die von einem Nicht-Schlüssel abhängen

Der Vollständigkeit halber erwähne ich noch die vierte Regel (jawohl, die gibt's auch). Ein Beispiel hierzu müßt Ihr Euch aber selbst ausdenken. Sie kommt in der Praxis eh selten (wenn überhaupt!) vor.

Die 4. Regel:

Die vierte Normalform verbietet jede Abhängigkeit zwischen Nicht-Primärschlüssel-Attributen. (???)

Vielleicht existieren weitere Regeln, bitte schreibt mir, falls sie einer kennt oder selbst formuliert hat.

Damit wären wir mit der Datennormalisierung fertig. Die Vorteile der getrennten Relationen sind klar erkennbar; alle Nachteile sind beseitigt.

- Pro Diskette können beliebig viele Programme erfaßt werden; das Löschen eines Programms ist nach Angabe von Disk-Nr. und Namen möglich.
- Für jedes Programm ist die Erfassung individueller Informationen möglich (z.B. Bedienung, Monitor, Preis usw.).
- Die Adresse des zuständigen Distributors ist sofort verfügbar.
- Die Gesamtsumme aller Programme ist gleich der Anzahl der Tupel in der Tabelle PROGRAMM.
- Die Ermittlung des insgesamt freien oder belegten Disk-Platzes geschieht über die Summe der entsprechenden Domäne in der Tabelle DISK.

Wie geht's weiter?

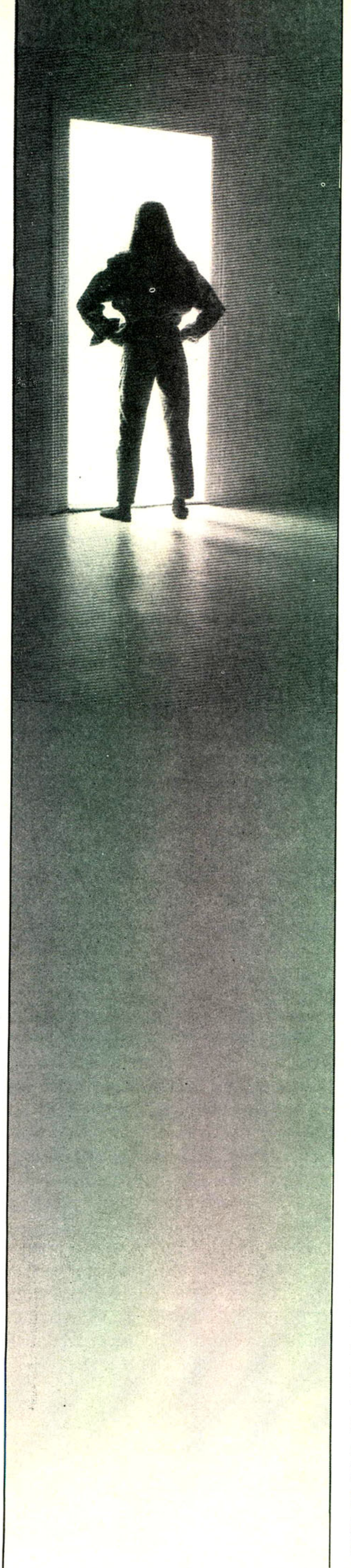
Bis zur nächsten Ausgabe werdet Ihr wohl die Schritte in der Entwicklung normalisierter Tabellen verstanden haben (mal ehrlich: so schwer ist es doch gar nicht!).

Ich werde im zweiten Teil meines Kurses endlich mit der praxisnahen Programmierung beginnen, und die Befehle zum Anlegen und Ändern einer Datenbank beschreiben. Bis dahin:

fröhliches Datennormalisieren !!

Paul Fischer

ENDE



AUF DER SCHWELLE ZUM LICHT

Laufwerksverwaltung

Diesmal widmen wir uns der Verwaltung der Laufwerke unter GEMDOS. Dabei wird, wie versprochen, der Drive-Media-Descriptor erklärt. Außerdem ist das Thema "Cluster- und Sektornumerierung" vom letzten Mal auch noch nicht ausgeschöpft.

Nachtrag

Doch zuerst noch ein kleiner Nachtrag zur FAT-Verwaltung. Es ist vielleicht noch ganz interessant, den Algorithmus zu kennen, mit dem ein freier Sektor in der FAT ausgewählt wird, wenn eine Datei verlängert wird.

Die Strategie ist eigentlich sehr einfach. Beim letzten Cluster der Datei beginnend, wird einfach der erste freie Cluster gesucht. Wenn dabei der letzte Cluster (nach Meinung des GEMDOS der letzte, eigentlich der drittletzte) erreicht wird, setzt man die Suche beim ersten Cluster (der Nummer 2) fort. Wenn 'numcl'- Einträge erfolglos untersucht wurden, ist das Speichermedium voll, und die Funktion bricht ab.

Nach dem Kreieren einer Datei beginnt die Suche für den ersten Cluster von vorne.

Anmeldung von Laufwerken

Damit GEMDOS mit den verschiedenen Massenspeicher-Laufwerken zusammenarbeiten kann, müssen diese bei ihm "angemeldet" sein. Dies geschieht aber nicht explizit dadurch, daß ein Treiberprogramm eine spezielle Funktion aufruft oder irgendwelche Systemvariablen manipuliert, sondern wird von GEMDOS automatisch durchgeführt, wenn das Laufwerk zum ersten Mal mittels einer GEMDOS-Funktion angesprochen wird.

GEMDOS führt Buch darüber, welche Laufwerke ihm schon bekannt sind. Dazu dient eine Bit-Maske ('drvmap'), wobei ein gesetztes Bit bedeutet, daß das Laufwerk angemeldet ist. Bit 0 ist dabei Laufwerk A: zugeordnet, Bit 1 gehört zu Laufwerk B:, usw.

Bei jeder GEMDOS-Funktion, die auf ein Laufwerk zugreifen muß (insbesondere also die Datei- und Disk-Funktionen F... bzw. D...), wird von einer Routine überprüft, ob der Zugriff auf das gewünschte Laufwerk möglich ist. Diese Routine wird im folgenden 'd_chkdrv' genannt.

Wenn das Laufwerk schon bekannt

ist, also das zugehörige Bit in 'drvmap' gesetzt ist, geht man davon aus, daß es auch ansprechbar ist. Man überprüft nur noch, ob für das Laufwerk schon ein Standardpfad existiert. Ist dies nicht der Fall, wird das Root Directory als aktuelles Directory festgelegt (dazu kommen wir noch einmal in einer späteren Folge). Ist das Laufwerk dagegen GEMDOS noch unbekannt, wird das BIOS mit 'Getbpb' um Hilfe gefragt. Wenn hier ein Fehler zurückgeliefert wird, ist entweder kein Treiber für das Laufwerk installiert oder es kann nicht angesprochen werden (z.B. wenn der Boot-Sektor der Diskette einen Lesefehler hat).

Im Normalfall gibt 'Getbpb' die Adresse des "BIOS-Parameter-Blocks" (BPB) zurück, an dessen Aufbau Sie sich vielleicht noch aus der letzten Folge erinnern. Der BPB wird nun ausgewertet und der "Drive Media Descriptor" aufgebaut (siehe nächsten Abschnitt).

Anschließend definiert wie oben beschrieben, das Root-Directory als aktueller Pfad.

Interessant ist in diesem Zusammenhang noch, daß 'd_chkdrv', wenn alles geklappt hat, die Laufwerkskennung an den Aufrufer zurückgibt. ERROR (-1L) zeigt einen eventuellen Fehler an. Wenn die aufrufenden Routinen diese Kennung und nicht die, die sie 'd_chkdrv' übergeben haben, für die eigentlichen Operationen benutzen würden, könnten in 'd_chkdrv' Laufwerke auf andere umgeleitet werden, bzw. es könnten "virtuelle Laufwerke" verwaltet werden.

An der Verwendung des Konjunktivs merken Sie schon, daß dies ein Wunschtraum bleibt, denn die Rückgabe von 'd_chkdrv' wird nicht weiter beachtet (außer daß die Fehlermeldung abgefragt wird).

Dieser in 'd_chkdrv' auftretende interne Fehler, nämlich daß ein Laufwerk nicht angesprochen werden kann, ist die in der Januar-Folge als "Laufwerk unbekannt" bezeichnete Ursache von GEMDOS-Fehlermeldungen (am besten nochmal nachlesen).

Der "Drive Media Descriptor"

Jeder eifrige Leser von Original ATARI-Dokumentationen ist schon über die Bezeichnung "Drive Media Descriptor" gestolpert: Bei der Angabe (von Dokumentation kann man hier wohl kaum sprechen) des "Buffer Control Blocks" (BCB) taucht dieser Begriff auf, ohne daß ein weiteres Wort über ihn verloren wird. Hier soll nun sein Geheimnis, das eigentlich gar keins ist, gelüftet werden.

Sein Aufbau ist in Abb. 1 dargestellt. Die Komponenten lassen sich am besten erläutern, indem man die Initialisierung des DMD bei der Anmeldung eines Laufwerks betrachtet.

```
typedef struct
{   int d_roff[3];   /* Sektornummer-Offsets für FAT,DIR,DATA   */
    int d_drive;     /* Laufwerkskennung 0..15                   */
    int d_fsiz;      /* FAT-Größe in Sektoren                    */
    int d_clsiz;     /* Cluster-Größe in Sektoren                */
    int d_clsizb;    /* Cluster-Größe in Bytes                   */
    int d_recsiz;    /* Sektor-Größe in Bytes                    */
    int d_numcl;     /* Zahl der Datencluster                    */
    int d_lclsiz;    /* 2er-Logarithmus von clsiz                */
    int d_mclsiz;    /* Bit-Maske für clsiz                      */
    int d_lrecsiz;   /* 2er-Logarithmus von recsiz               */
    int d_mrecsiz;   /* Bit-Maske für recsiz                     */
    int d_lclsizb;   /* 2er-Logarithmus von clsizb               */
    FD *d_fatfd;     /* Zeiger auf FD der FAT                    */
    long d_dummy;    /* unbenutzt                                */
    DD *d_rdd;       /* Zeiger auf DD des Root Directorys        */
    int d_flag;      /* FAT-Typ: 0: 12-Bit, 1: 16-Bit            */
} DMD;
```

Abb. 1 : Drive Media Descriptor (DMD)

Wie der Name schon sagt, beschreibt der DMD ein Laufwerk bzw. das damit verbundene Speichermedium. Er erlaubt GEMDOS, die Kommunikation mit den unterschiedlichsten Laufwerken und Medien, da er recht flexibel ist. Wie letzten Monat schon deutlich wurde, ist dies aber alles nur graue Theorie, da hier so manches nicht funktioniert.

'd_drive' enthält, wie man sich beinahe denken kann, die Kennung des Laufwerks, das der DMD beschreibt.

Einige Werte im DMD werden direkt aus dem BPB übernommen, was durch die ähnliche Namensgebung angedeutet werden soll. So ist 'd_clsiz' die Cluster-Größe in Sektoren (wie 'b_clsiz'), 'd_clsizb' gibt die Cluster-Größe in Bytes an (wie 'b_clsizb'), 'd_recsiz' ist die Sektor-Größe in Bytes (wie 'b_recsiz'), 'd_fsiz' ist die FAT-Größe in Sektoren (wie 'b_fsiz') und 'd_numcl' ist die Anzahl der Daten-Cluster (wie 'b_numcl'). 'd_flag' enthält eine Kopie des Bit 0 von 'b_flags[0]', gibt also Auskunft, ob eine 12-oder 16-Bit-FAT vorliegt. Die anderen Bits von 'd_flag' sind immer Null, unabhängig von 'b_flags[0]'.

Bei der Berechnung von Datei-Positionen, Sektor-Nummern, usw. rechnet GEMDOS fleißig mit diesen Werten herum. Dabei ist es oft notwendig, durch sie zu dividieren bzw. bestimmte Bits auszublenden. Dies wird mit Shift-Operationen und Logisch-AND-Befehlen erreicht. Damit man die hierbei benötigten Shift-Operanden (Zweierlogarithmen) und Bit-Masken nicht jedesmal neu berechnen muß, merkt sich GEMDOS einige davon im DMD.

In 'd_lclsiz' wird der Zweierlogarithmus von 'd_clsiz' gespeichert, 'd_mclsiz' ist eine Bit-Maske zum Ausblenden niederwertiger Bits (Tab. 1). Die entsprechenden Werte für 'd_recsiz' sind in 'd_lrecsiz' und 'd_mrecsiz' abgelegt (wer hätte das

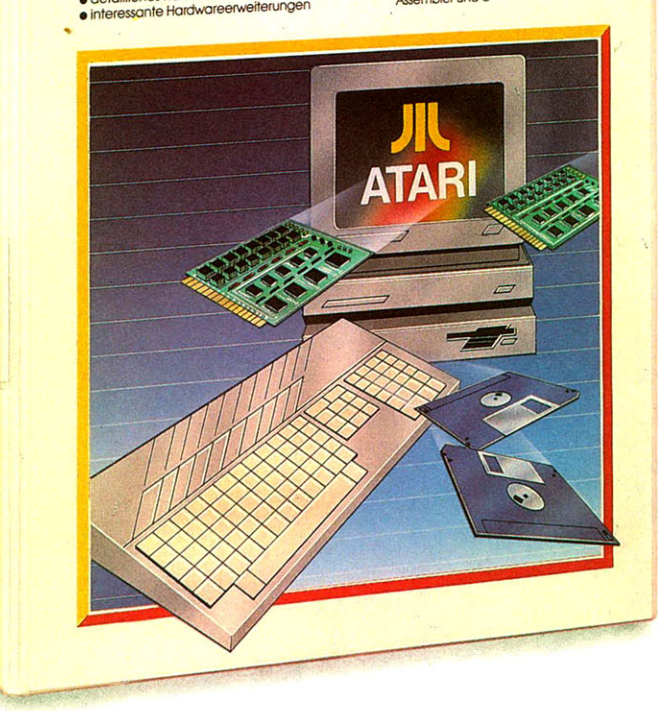
'clsiz' bzw. 'recsiz'	'lclsiz' bzw. 'lrecsiz'	'mclsiz' bzw. 'mrecsiz'
1	0	$0000
2	1	$0001
4	2	$0003
:	:	:
512	9	$01FF
1024	10	$03FF
:	:	:

Tab. 1 : Shift- und Maskier-Werte

Das Tuning-Programm für Ihren Atari ST

Interessiert an 68000er-Programmierung in Assembler und C?
Blättern Sie bitte um!

Erst prüfen, dann kaufen
Schauen Sie sich dieses Werk in Ruhe an: 10 Tage lang dürfen Sie Ihr Ansichtsexemplar unverbindlich zu Hause prüfen.



Dieses neue Nachschlagewerk bietet Ihnen

■ **ausführliche Beschreibungen des äußeren und internen Aufbaus der Atari ST-Reihe:** Sie lernen die einzelnen Bausteine wie 68000er-Prozessor, DMA-Controller oder Glue-Baustein im Detail kennen und erfahren, wie diese Komponenten zusammenarbeiten;

■ **das Know-how zur Systemprogrammierung:** Anhand von Beispielen werden Sie mit GEM und den TOS-Komponenten GEM-DOS, BIOS und XBIOS vertraut. Sie lernen Systemroutinen, z. B. für die Mausprogrammierung oder Fenstermanagement, zu nutzen. Den detaillierten Betriebssystemübersichten entnehmen Sie u. a. die Speicheradressen der verschiedenen TOS-Versionen oder den Befehlsvorrat an GEM-, AES- und GEM-VDI-Routinen;

■ **Kurse für erfolgreiches Softwareengineering:** Mit Beispielen aus zentralen Anwendungen wie Grafik oder Dateiverwaltung verfügen Sie gleichzeitig über sofort einsetzbare Lösungen;

■ **Programmiersprachkurse** für C, 68000er-Assembler, GFA-Basic;

■ **Systemtuning durch Hardwareerweiterungen und Bausteinprogrammierung:** Präzise Anleitungen zeigen Ihnen detailliert, wie Sie bei Ihrem Atari eine RAM-Erweiterung oder eine ROM-TOS-Aufrüstung vornehmen. Bauanleitungen mit Platinenlayouts auf Folie, zusammen mit der erforderlichen Betriebssoftware, ermöglichen Ihnen den kostengünstigen Selbstbau eines hochleistungsfähigen Festplattenlaufwerks, eines EPROMers sowie eines universellen Erweiterungsports.

■ **praxiserprobte Musterlösungen** für **Technik/Mathematik** (Logikentwicklungssimulator zur Entwicklung von digitalen Schaltungen, Fouriertransformationen, Matrizenmultiplikation u. a.), **Planung** (z. B. Netzplantechnik), **Grafik** (z. B. Berechnung von Zentral- und Parallelprojektionen, 3D-Grafik, Bilder in 512 Farben bei gleichzeitigem Maus- und Tastaturhandling), **Sound** (Sound-Sampler u. a.) **Add-On-Programme zur Standardsoftware, Desktop-Publishing;**

■ **Tips, Tricks und Utilities:** Sie erhalten **Hardwaretips** (z. B. für den Einsatz eines 5¼"-Laufwerks), **Floppyroutinen** u. a. Datenübernahme von MS-DOS-Disketten), **Druckerutilities** (Ausdruck des Directory), **Systemutilities** (RAM-Disk, EPROM-Programmierung, Autoboot mit integriertem Monitorschutz), **Konvertierutilities** (zum problemlosen Datenaustausch zwischen Atari ST und Amiga); **Komprimierroutinen für Programme;**

■ **den preisgünstigen Diskettenservice**

■ **regelmäßige Ergänzungsausgaben zum Grundwerk** mit aktuellen Informationen rund um den Atari wie z. B. neuentwickelte Hard- und Software, neue Betriebssystemversionen.

Fordern Sie noch heute mit nebenstehender Bestellkarte an:

Erfolgreiches Hard- und Softwaretuning der Atari ST 260/520/1040 und Weiterentwicklungen

Stabiler Ringbuchordner, Format DIN A 4, Grundwerk ca. 400 Seiten, Bestell-Nr. 3700, Preis: DM 92,–

Alle 2–3 Monate erhalten Sie Ergänzungsausgaben zum Grundwerk mit je ca. 120 Seiten zum Seitenpreis von 38 Pfennig (Abbestellung jederzeit möglich).



INTEREST-VERLAG
Fachverlag für anspruchsvolle Freizeitgestaltung
Industriestraße 21
D-8901 Kissing
Tel. 08233/2101-0

Bitte abtrennen



Ja, senden Sie mir bitte sofort

☐ Expl. **Erfolgreiches Hard- und Softwaretuning der Atari ST 260/520/1040 und Weiterentwicklungen**
Stabiler Ringbuchordner, Format DIN A 4, Grundwerk ca. 400 Seiten, Bestell-Nr. 3700, Preis: DM 92,–



☐ Expl. **„Erfolgreiches Programmieren von 68000er-Systemen in Assembler und C"**
stabiler Ringbuchordner, Format DIN A 4, Grundwerk ca. 400 Seiten, Bestell-Nr. 3400, Preis: DM 92,–



Zu jedem dieser Werke erhalten Sie alle 2–3 Monate Ergänzungsausgaben zum Grundwerk mit je ca. 120 Seiten zum Seitenpreis von 38 Pfennig (Abbestellung jederzeit möglich).

Unterschreiben Sie hier bitte Ihre Bestellung!
Bei Minderjährigen ist die Unterschrift eines gesetzlichen Vertreters erforderlich. Ohne Ihre Unterschriften kann die Ansichtsbestellung nicht bearbeitet werden.



Datum Unterschrift

Bitte unterschreiben Sie auch Ihre Sicherheitsgarantie!
mit der Sie folgendes zur Kenntnis nehmen: Sie haben das Recht, Ihr angefordertes Werk innerhalb von 10 Tagen ab Lieferung an die Bestelladresse zurückzusenden, wobei für die Fristwahrung das Datum der Absendung genügt. Sie kommen dadurch von allen Verpflichtungen aus der Bestellung frei.




Datum Unterschrift

68000er-Systeme perfekt programmieren in Assembler und C



Erst prüfen, dann kaufen
Schauen Sie sich dieses Werk in Ruhe an: 10 Tage lang dürfen Sie Ihr Ansichtsexemplar unverbindlich zu Hause prüfen.

Nutzen Sie die Stärken Ihres Atari!
Jetzt hilft Ihnen dieses neue Nachschlagewerk,
■ effiziente Problemlösungen auf **Betriebssystemebene** zu realisieren
■ anspruchsvolle **Anwenderprogramme** zu entwickeln
■ mit ausgefeilter **Bausteinprogrammierung** eine neue Atari-Dimension zu erschließen.

Die wichtigsten Themen auf einen Blick:
■ **detaillierte Hardware-Beschreibungen** der Prozessorfamilie 68000 und ihrer 8-, 16-, 32-Bit-Peripheriebausteine mit technischen Kennwerten, Pinbelegung, internen Architekturen, Befehlsbeschreibungen;
■ **Betriebssysteme:** Sie lernen die Strukturen der 68000er-Betriebssysteme (wie GEM-TOS. OS-9 und Unix) kennen und erfahren, wie Sie die Systemroutinen zur Optimierung Ihrer Assembler- und C-Programme nutzen;
■ **Softwareengineering:** die optimale Vorgehensweise von der Problemanalyse über die Codierung bis hin zur abschließenden Dokumentation;
■ **Programmierkurse für effiziente Anwender-, System- und Bausteinprogrammierung in Assembler und C.** Damit verfügen Sie zugleich über sofort einsetzbare Routinen, z. B. zur Drucker- und Bildschirmansteuerung;
■ **die C-Compiler-Werkstatt:** Schritt für Schritt programmieren Sie selbst einen C-Compiler inkl. Funktionsbibliotheken;
■ **Assembler-Makrobibliotheken, Anwender- und Funktionsbibliotheken in C;**
■ **Tips und Utilities** wie Schnittstellen- und Backup-Routinen, Fensterverwaltung;
■ **bewährte Musterlösungen** wie zu Rechnerkopplung und Multitasking;
■ **Anleitung für raffiniertes System-tuning,** z. B. durch Einsatz schnellerer Peripheriebausteine/Höhertakten der CPU;
■ **Praxiswissen zu speziellen Einsatzbereichen** wie Messen, Steuern, Regeln;

Dieses Werk veraltet nie
Unsere 68000er-Fachredaktion versorgt Sie regelmäßig und zuverlässig mit aktuellen Informationen über Hardware- und Betriebssystem-Weiterentwicklungen, weiteren Utilities und Musterlösungen sowie Bibliothekserweiterungen in Assembler und C.

Bitte abtrennen

Meine Anschrift:

Name

Vorname

Straße, Haus-Nr.

PLZ/Ort

Nutzen Sie unseren kostenlosen Informations-Service:

☐ Grafik und CAD
☐ Desktop-Publishing
☐ dBase-Programmierung
☐ Messen, Steuern, Regeln mit dem Atari ST
☐ Aktuelles IC-Datenbuch

IN 160050

Postkarte/Antwort

60 Pfennig die sich lohnen!

INTEREST-VERLAG
Fachverlag für anspruchsvolle Freizeitgestaltung
Herrn Michel
Industriestraße 21

D-8901 Kissing

Fordern Sie noch heute mit nebenstehender Bestellkarte an:

Erfolgreiches Programmieren von 68000er-Systemen in Assembler und C

stabiler Ringbuchordner, Format DIN A 4, Grundwerk ca. 400 Seiten. Bestell-Nr. 3400, Preis: DM 92,–

Alle 2–3 Monate erhalten Sie Ergänzungsausgaben zum Grundwerk mit je ca. 120 Seiten zum Seitenpreis von 38 Pfennig (Abbestellung jederzeit möglich).





Industriestraße 21
D-8901 Kissing
Tel. 08233/2101-0

	Standard-Diskette	fiktiv	fiktiv
CLSIZ	2	1	4
datrec	18	11	50
fatrec	6	3	21
d_roff[2]	14	9	42
RDLEN	7	5	9
RDCL	4	5	3
RD-Clusternr.	-5.. -2	-6.. -2	-4.. -2
RD-Sektornr.	-10.. -4	-6.. -2	-16.. -8
d_roff[1]	21	12	57
FSIZ	5	3	20
FCL	3	3	5
FAT-Clusternr.	-8.. -6	-9.. -7	-9.. -5
FAT-Sektornr.	-16..-12	-9.. -7	-36..-17
d_roff[0]	22	12	57

Tab. 2 : Beispiele zur GEMDOS-Sektor-Zählung

gedacht?). Die Bit-Masken sind zwar gerade die um eins verminderten Werte selbst, aber GEMDOS holt sie sich extra aus einer Tabelle ('f_masks', s.u.)!

Zur Verwaltung von Dateien (einschl. Directories und FAT) gibt es weitere Strukturen ("File Descriptoren" und "Directory Descriptoren"), um die wir uns ein andermal kümmern. Hier sei nur erwähnt, daß sie zusammen mit dem DMD ein kompliziertes Netz bilden, welches vom DMD aus gut "aufgerollt" werden kann.

Beim Einrichten des DMD werden die File Descriptoren für das Root Directory und die FAT sowie der Directory Descriptor des Root Directories ebenfalls initialisiert.

Übrig bleiben nun noch die 'd_roff[0..2]'-Werte. Dabei handelt es sich um die letzten Monat erwähnten Sektornummer-Offsets für FAT-, DIR- und DATA-Sektoren. Die GEMDOS-Sektornummer plus entsprechendem Offset ergibt die BIOS-Sektornummer.

In der letzten Folge waren die beiden Zählweisen einander gegenübergestellt, wobei auffiel, daß die GEMDOS-Sektornummern nicht durchgehend waren, sondern daß es merkwürdige Lücken in der Numerierung gab. Heute soll der Algorithmus erklärt werden, mit dem diese Zählweise festgelegt wird.

Bisher habe ich Ihnen eine weitere Komplikation noch verschwiegen. Obwohl eigentlich nur Daten-Sektoren zu Clustern zusammengefaßt werden, gibt es auch Cluster-Nummern für Root Directory (RD) und FAT. Das liegt daran, daß RD und FAT intern wie Dateien behandelt und daher bei der Berechnung von Dateipositionen (Seek-Funktion) wie bei Dateien Cluster in Sektoren umgerechnet werden. Da es in Wirklichkeit aber gar keine Cluster gibt, müssen Cluster- und Sektornumerierung so angelegt sein, daß die Pseudo-Cluster-Nummern eindeutig auf GEMDOS-Sektornummern abgebildet werden.

Zum besseren Verständnis der etwas komplizierten Zusammenhänge dient Tab. 2, wo Beispiele für die nun folgenden Erklärungen angegeben sind. Aus der Länge des RD in Sektoren ('b_rdlen') errechnet sich seine Länge in Clustern ('rdcl'). Da die Zahl der RD-Sektoren kein Vielfaches der Clustergröße sein muß, wird diese Clusterzahl aufgerundet. Die genaue Formel lautet:

$$rdcl = \frac{rdlen + clsiz - 1}{clsiz}$$

(Der bei der Division auftretende Rest wird ignoriert). Der erste "Cluster" des RD ist 'rdst':

$$rdst = -1 - rdcl$$

Das RD belegt damit die Clusternummern 'rdst' bis -2; die Clusternummer -1 existiert nicht, ebenso die Cluster 0 und 1. Die erste GEMDOS-Sektornummer ergibt sich durch Multiplikation mit der Anzahl der Sektoren pro Cluster. Durch die Aufrundung bei der Berechnung von 'rdcl' ergeben sich in der "Ausnutzung" der GEMDOS-Sektornummern Lücken (s. Beispiele in Tab. 2).

Entsprechend verfährt man mit der FAT:

$$fcl = \frac{fsiz + clsiz - 1}{clsiz}$$

$$fst = rdst - fcl$$

Die FAT-Cluster liegen unmittelbar vor den RD-Clustern, denn sie haben die Nummern 'fst' bis 'rdst'-1. Auch hier ergibt sich die erste GEMDOS-Sektornummer durch Multiplikation von 'fst' mit 'clsiz'.

Nachdem nun festgelegt wurde, welchen Pseudo-Clustern die RD- und FAT-Sektoren zugeordnet werden, kann GEMDOS die gesamte Verwaltung wie bei User-Dateien erledigen. Beim tatsächlichen Zugriff werden nun die GEMDOS-Sektornummern, mit denen sich die Sektorpuffer-Verwaltung herumschlagen muß, mit den 'd_roff'-Offsets zu BIOS-Nummern zurückgerechnet. Diese Offsets müssen nun so berechnet werden, daß diese Rücktransformation klappt.

Für Daten-Sektoren:

$$d_roff[2] = datrec - (2 * clsiz)$$

'datrec' ist die aus dem BPB entnommene BIOS-Nummer des ersten Daten-Sektors. Da der erste Cluster immer die Nummer 2 hat, ergeben sich für Daten-Sektoren die Nummern ab 'datrec' aufwärts.

Für FAT-Sektoren:

$$d_roff[0] = fatrec - (fst * clsiz)$$

'fatrec' ist die ebenfalls im BPB gefundene BIOS-Nummer des ersten Sektors der zweiten FAT, die, wie Sie seit einem Monat wissen, eigentlich die erste FAT ist. Zu beachten ist, daß 'fst' negativ ist, und damit 'd_roff[0]' größer als 'fatrec' wird. Dies ist aber korrekt, da die bei der Umrechnung addierten GEMDOS-Sektornummern negativ sind, so daß sich BIOS-Nummern ab 'fatrec' aufwärts ergeben.

geht weiter...

Für RD-Sektoren:

d_roff[1] =
*fatrec + fsiz - (rdst * clsiz)*

Das RD muß unmittelbar nach der zweiten FAT liegen, beginnt also bei 'fatrec' plus 'fsiz' ('fsiz' ist die Zahl der Sektoren pro FAT, zu finden im BPB - wo wohl sonst?). Der Rest ist wie bei der FAT.

So, nun wissen Sie hoffentlich alles, was man über Sektor- und Clusternummern wissen muß. Bei den bisherigen Erfahrungen mit GEMDOS muß man sich wundern, daß das ganze Hin- und Hergerechne so reibungslos funktioniert; zumindest ist mir in dieser Hinsicht noch kein Fehler bekannt.

Noch eine Ergänzung für diejenigen, die den Artikel über die Speicherverwaltung kennen:

Die DMDs werden im "internen GEMDOS-Speicher" abgelegt. Da es hiervon aber nur relativ wenige gibt, ist die Anzahl der Laufwerke selbst nicht so entscheidend. Allerdings wird man bei vielen Laufwerken auch viele Ordner ansprechen, so daß die Chance, daß einem der GEMDOS-Speicher ausgeht, bei jedem neuen Laufwerk steigt (eine zweite Floppy wirkt hier schon Wunder).

Da ein DMD 42 Byte groß ist, was aufgerundet drei 8-Wort-Einheiten (3*8*2 = 48) sind, finden sich freigegebene DMDs in der 'mifl'-Liste 3. Der DMD ist übrigens die einzige Struktur, die über die "3er-Liste" verwaltet wird.

Abmelden von Laufwerken

GEMDOS erkennt zwar selbständig neue Laufwerke und richtet dafür die internen Datenstrukturen ein, doch ist die Anmeldung eines Laufwerks nicht vorgesehen.

Aber auch hierfür gibt es einen "automatischen" Mechanismus, nämlich den Mediumwechsel. Bei einem durch das BIOS gemeldeten Mediumwechsel werden zuerst alle Datenstrukturen des betroffenen Laufwerks freigegeben. Dies sind der DMD, die File und die Directory-Descriptoren. Weiterhin werden alle Pfade zu Directories sowie die Sektor-Puffer für ungültig erklärt (siehe hierzu auch die erste und zweite Folge).

Anschließend wird mit der BIOS-Funktion 'Getbpb' überprüft, ob das Laufwerk (mit einem anderen Medium) überhaupt noch verfügbar ist. Ist dies der Fall, werden die Strukturen zur Disk-Verwaltung neu initialisiert, so daß das Laufwerk GEMDOS wieder bekannt ist.

Wenn das Laufwerk nicht mehr ansprechbar ist, wird es für GEMDOS abgemeldet, indem das entsprechende Bit von 'drvmap' gelöscht wird. Einer erneuten Anmeldung durch Zugriff steht nichts im Wege.

Dieses Verfahren kann man sich zu Nutze machen, wenn man gezielt einzelne Laufwerke abmelden will. Dies kann sinnvoll sein, wenn das Laufwerk unter verschiedenen Kennungen angesprochen werden soll. Würde die Abmeldung nicht erfolgen und der Treiber einfach auf seine alte Kennung nicht mehr reagieren, würden die Datenstrukturen als "Leichen" im Speicher herumliegen und kostbaren internen Speicher verbrauchen.

Außerdem kann ein simulierter Mediumwechsel bei allen Laufwerken einfach dazu dienen, internen Speicher freizugeben, was nach ausgiebigem Öffnen von vielen Ordnern empfehlenswert ist. Dazu läßt sich eine solche Routine leicht in ein Accessory einbauen.

Listing 1 zeigt nun, wie man's macht. In der Funktion 'free_drv' werden zunächst die Vektoren der elementaren Disk-Routinen 'Mediach', 'Getbpb' und 'Rwabs' auf eigene Routinen umgesetzt. Durch den Aufruf der GEMDOS-Funktion 'Dfree' (andere tun's auch) wird GEMDOS zu einem Zugriff auf das Laufwerk gezwungen. Die eigenen Disk-Routinen geben nun Meldungen zurück, wie sie bei einem Diskettenwechsel auftreten.

Die Funktion 'free_all' generiert Diskettenwechsel für alle 16 möglichen Laufwerke.

Da die Disk-Vektoren im geschützten Speicherbereich liegen, darf auf sie nur im Supervisor-Mode zugegriffen werden. Da man dieses Problem als "Betriebssystem-Hacker" öfter hat, sind die allgemeinen "Peek & Poke"-Routinen für solche Dinge ganz brauchbar. Wer erinnert sich hier nicht an seine alten 8-Bitter, bei denen die Kenntnis geheimnisvoller "Poke-Befehle" unerläßlich war?

Bei der Anwendung dieses kleinen Programms sollten Sie darauf achten, daß auf den betroffenen Laufwerken keine Dateien mehr geöffnet sind, da diese bei dem "Mediumwechsel" nicht mehr ordnungsgemäß geschlossen werden können, und daher Datenverlust droht. Ferner gehen alle Standardpfad-Einstellungen verloren.

Systemvariable 'drvbits'

Wie wir nun wissen, steht in der GEMDOS-Variablen 'drvmap', welche Laufwerke bekannt sind. Andererseits gibt es noch die BIOS-Systemvariable 'drvbits' ($4C2). 'drvbits' ist ebenfalls eine Bit-Maske für vorhandene Laufwerke (sogar 32 Bit!), hat aber nichts mit GEMDOS zu tun.

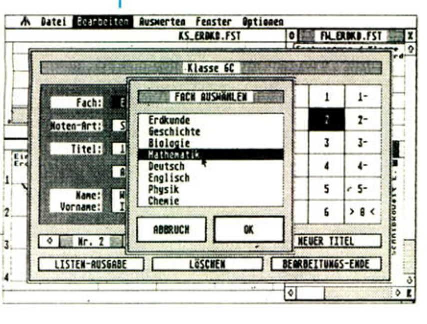
Während 'drvmap' angibt, welche Laufwerke schon einmal angesprochen wurden, steht in 'drvbits', ob das Laufwerk überhaupt verfügbar ist. Massenspeicher-Treiber sollten allerdings die von ihnen belegten Laufwerkskennungen hier eintragen. 'drvbits' wird nämlich von der BIOS-Funktion 'Drvmap' zurückgeliefert, damit Anwenderprogramme feststellen können, welche Laufwerke ansprechbar sind. Vom AES wird diese Möglichkeit während der Reset-Phase auch ausgiebig genutzt.

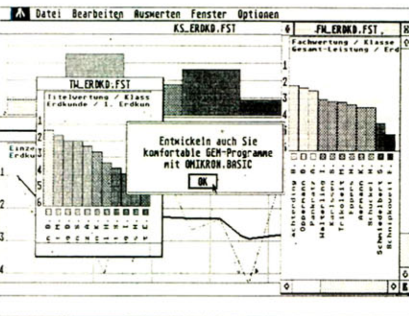
GEMDOS-Variablen für Disk-Verwaltung

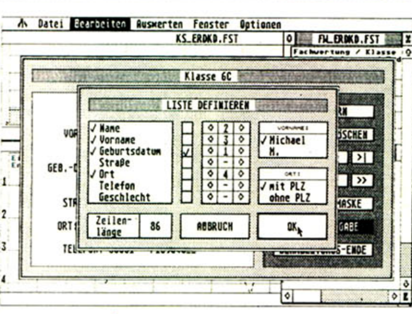
Die globalen GEMDOS-Variablen für die Disk-Verwaltung sind in Abb. 2 dargestellt, zusammen mit denen für die Sektor-Pufferung, die ich letztes Mal vergessen habe. Die erste

OMIKRON.BASIC: „Sieger aller Klassen“

(68000er Magazin, Oktober 1987, Seite 57)







eine gute Nachricht für alle GFA-Basic-Benutzer: OSKAR* wandelt Ihre Programme in OMIKRON.BASIC-Programme um.

„Der OMIKRON.Basic-Interpreter stellt zusammen mit dem Compiler das beste Basic-Entwicklungssystem für den ATARI ST dar.“ (ST-Computer 9/87, Seite 65)

* OSKAR ist ein Programm der Markt & Technik Verlag AG. Sie finden es im Happy-Computer-Sonderheft „St-Magazin“, Nr. 27.

ARITHMETIK: Rechengenauigkeit bis 19 Stellen bei allen Funktionen ● Rechenbereich bis 5.11 E±4931

GEM: Komplette GEM-Library ● Sämtliche AES- und VDI-Funktionen direkt mit Namen verfügbar ● Eigene BITBLIT-Routine

EXTRAS: Masken-INPUT ● SORT-Befehl sortiert beliebige Felder, auch mit Umlauten ● Matrizenbefehle

STRUKTUR: Prozeduren und mehrzeilige Funktionen mit Übergabe- und Rückgabe-Parametern und lokalen Variablen ● REPEAT... UNTIL, WHILE... WEND, mehrzeiliges IF... THEN... ELSE... ENDIF ● Labels bei GOTO, GOSUB, ON GOTO und ON GOSUB

EDITOR: Mit oder ohne Zeilennummern (umschaltbar) ● Drei Schriftgrößen bis 57 x 128 Zeichen ● Frei definierbare Teach-In-Funktionstasten

GESCHWINDIGKEIT: FIT-Code (FIT = Fast Interpreting Technique) ● Wir kennen keine schnelleren 68000-Fließkomma-Routinen ● Volle Integer-Arithmetik ● Eigene Disk-Routinen für beschleunigten Dateizugriff

KOMPATIBILITÄT: Für alle ATARI ST (auch MEGA ST) ● 99 % MBASIC-kompatibel ● Editor findet Inkompatibilitäten, dadurch einfachste Anpassung

● Diskettenversion mit Demodiskette und Handbuch **DM 179,–**

● Modulversion mit Demodiskette und Handbuch **DM 229,–**

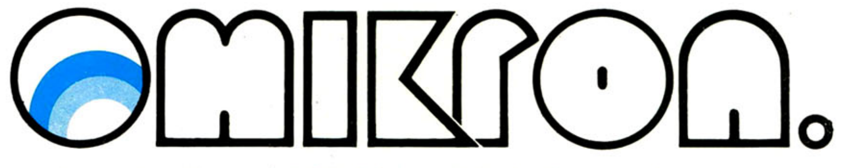
COMPILER

Noch mehr Tempo erreichen Ihre mit OMIKRON.BASIC entwickelten Applikationen durch unseren OMIKRON.BASIC-COMPILER.

Er erlaubt es Ihnen, bequem in Basic interpretativ zu programmieren, um nach der Compilierung Geschwindigkeiten zu erreichen, die bisher C-Compilern vorbehalten waren. Der Aufbruch in neue Dimensionen der Basic-Programmierung hat begonnen.

● Diskette mit Anleitung **DM 179,–**





... denn das Beste ist für Ihren ATARI ST gerade gut genug!

OMIKRON.SOFTWARE

Erlachstraße 15 · D-7534 Birkenfeld · ☎ (0 70 82) 53 86

Frankreich: OMIKRON · France, 11 Rue Dérodé, F-51100 Reims
Luxemburg: ELECOMP, 11 Avenue de la gare, L-4131 Esch-Alzette
Niederlande: Terminal Software Publicaties, Postbus 111, NL-5110 Baarle-Nassau
Österreich: Ueberreuter Media, Laudengasse 29, A-1082 Wien
Schweiz: Thali AG, Industriestraße 6, CH-6285 Hitzkirch

```
BCB bcbx[4];            /*  $60a4/  $8846: BIOS: 4-Standard-BCBs */
char secbuf[4][512];  /*  $4e22/  $75c4: BIOS: 4 Puffer für je
                            1 Sektor */

int drvmap;             /*  $5fe2/  $8784: Bit-Map für angemeldete
                            Drives */
DMD *dmdx[16];          /*  $5bde/  $8780: DMD-Zeiger für alle Drives */
int f_masks[]           /*  $fdlbf6/$fd3022: Masken für
                            Datei-Operationen */
  = { 0x0000, 0x0001, 0x0003, 0x0007,
      0x000f, 0x001f, 0x003f, 0x007f,
      0x00ff, 0x01ff, 0x03ff, 0x07ff,
      0xffff, 0x1fff, 0x3fff, 0x7fff
    };
```

Abb. 2 : GEMDOS-Variablen zur Puffer- und Disk-Verwaltung

Adresse gilt für das alte TOS (6.2.1986), die zweite für das Blitter-TOS. Überflüssig zu erwähnen, daß diese Adressen nicht dokumentiert sind und nicht in Programmen verwendet werden sollten.
'bcbx' und 'secbuf' sind die vom BIOS eingerichteten 4 BCBs mit den dazugehörigen Sektor-Puffern. Sie sind legal über die Pufferlisten ('bufl[]') zugänglich.
'drvmap' wurde oben schon erläutert. GEMDOS hat eine Tabelle ('dmdx'), in der für jedes Laufwerk der Zeiger auf den DMD steht. Wenn ein Laufwerk unbekannt ist, enthält die Tabelle einen Nullzeiger (0L). Gesetzt bzw. gelöscht werden diese Zeiger bei der automatischen An- und Abmeldung der Laufwerke.
'f_masks' ist eine Tabelle mit Bit-Masken, die für Berechnungen verschiedenster Art benötigt werden. Bei der Anmeldung von Laufwerken ergeben sich die Werte für 'b_mrecsiz' und 'b_mclsiz' aus dieser Tabelle durch Indizierung mit den log2-Werten ('b_lrecsiz', 'b_lclsiz').
Im folgenden wird die Struktur des Prozeßdescriptors noch gebraucht werden, daher ist sie für diejenigen unter Ihnen, die den Artikel "Programmverwaltung" nicht gelesen haben, in Abb. 3 noch einmal dargestellt. Für die genaue Erklärung muß ich jedoch leider wieder auf das ST-Extra-Heft verweisen.

Mehr als 16 Laufwerke?

GEMDOS kann in seiner derzeitigen Version entgegen sämtlicher Gerüchte maximal 16 Laufwerke verwalten. Dies scheint viel zu sein, doch gibt es Leute, denen das noch zu wenig ist. Die Gründe für die Beschränkung auf 16 Laufwerke sind die folgenden:
'drvmap' ist nur eine 'int'-Variable (16 Bit). Durch eine einfache Änderung im GEMDOS-Quelltext könnte dies auf 'long' (32 Bit) geändert werden.
Das 'dmdx'-Feld müßte für 32 Laufwerke erweitert werden (für die Atari-Entwickler ebenfalls kein Problem).
Das 'p_drvx'-Feld im Prozeßdeskriptor (PD), in dem Handles für die Standardpfade abgelegt sind (dazu kommen wir in einer späteren Folge), ist für 16 Laufwerke dimensioniert. Da danach aber noch Platz im PD ist, wäre eine Erweiterung auf 32 Laufwerke ohne Änderungen möglich.
Allerdings gibt es Routinen, die alle Laufwerke aus 'p_drvx' in einer Schleife durchgehen (z.B. bei der Vererbung an Tochterprozesse). Hier müßte überall der Schleifenendwert auf 32 statt 16 gesetzt werden.
Bei Benutzung so vieler Laufwerke müßte aber auch der interne Speicher nochmals vergrößert werden, da GEMDOS hier ganz schön zuschlägt.

GEMDOS-Funktionen (TRAP #1)

Hier möchte ich noch kurz auf die GEMDOS-Funktionen eingehen, die mit der Disk-Verwaltung zusammenhängen.

Funktion $0e Dsetdrv

long Dsetdrv(int drv)

Das Default-Laufwerk wird auf 'drv' gesetzt (0,1.. für A:, B:, usw.). Dieses Laufwerk wird angesprochen, wenn in Pfadnamen keine Laufwerkskennung explizit angegeben ist.

Rückgabewert

Es wird eine 32-Bit-Maske der angemeldeten Laufwerke zurückgegeben (Bit 0,1,.. für A:,B:,..). Dabei handelt es sich jedoch nicht um die GEMDOS-Variable 'drvmap', sondern um die oben erwähnte BIOS-Systemvariable 'drvbits'.

```
typedef struct
{  char *p_lowtpa;      /* Zeiger auf Beginn des reservierten Bereichs */
   char *p_hitpa;       /* Zeiger auf Ende des reservierten Bereichs   */
   char *p_tbase;       /* Zeiger auf TEXT-Segment                     */
   long p_tlen;         /* Länge TEXT-Segment                          */
   char *p_dbase;       /* Zeiger auf DATA-Segment                     */
   long p_dlen;         /* Länge DATA-Segment                          */
   char *p_bbase;       /* Zeiger auf BSS-Segment                      */
   long p_blen;         /* Länge BSS-Segment                           */
   DTA *p_dta;          /* Zeiger auf DTA-Puffer                       */
   PD *p_parent;        /* Zeiger auf PD des Parent-Prozesses          */
   long res1;           /* nicht benutzt                               */
   char *p_env;         /* Zeiger auf Environment-String               */
   char p_devx[6];      /* Handles für Standard-Devices                */
   char res2;           /* nicht benutzt                               */
   char p_defdrv;       /* aktuelles Laufwerk (Default-Laufwerk)       */
   char res3[8];        /* nicht benutzt                               */
   char p_drvx[16];     /* Pfad_Handles für alle Laufwerke             */
   char res4[0x18];     /* nicht benutzt                               */
   long p_d0;           /* Prozessorregister D0                        */
   long p_a3;           /* Prozessorregister A3                        */
   long p_a4;           /* Prozessorregister A4                        */
   long p_a5;           /* Prozessorregister A5                        */
   long p_a6;           /* Prozessorregister A6                        */
   long *p_reg;         /* Zeiger auf restliche Prozessor-Register     */
   char p_cmdlin[128];  /* Kommando-Zeile (übergebener Parameter)      */
} PD;
```

Abb. 3: Der Prozeßdescriptor (PD)

```
typedef  struct
{   long b_free;        /* freie Cluster               */
    long b_total;       /* Gesamtzahl der Cluster      */
    long b_secsiz;      /* Sektor-Größe in Byte        */
    long b_clsiz;       /* Cluster-Größe in Sektoren   */
}  DISKINFO;
```

Abb. 4 : DISKINFO-Struktur

Arbeitsweise

GEMDOS merkt sich das aktuelle Laufwerk im PD als 'p_defdrv'. Dies bedeutet, daß jeder Prozeß sein eigenes aktuelles Laufwerk hat.

Die Funktion 'Dsetdrv' besteht nur aus zwei 'C'-Befehlen. 'drv' wird direkt nach 'p_defdrv' des PD kopiert. Dabei erfolgen keinerlei Kontrollen. Wenn das Laufwerk gar nicht existiert, wird dies frühestens beim versuchten Zugriff bemerkt (in der oben erläuterten Routine 'd_chkdrv'). Als zweites wird die BIOS-Funktion 'Drvmap' aufgerufen, deren Ergebnis direkt an den Aufrufer von 'Dsetdrv' zurückgegeben wird.

Bei der Gelegenheit gleich noch ein Wort zu unerlaubten Laufwerkskennungen: GEMDOS bemerkt zwar früher oder später, wenn ein Laufwerk nicht ansprechbar ist. Es reagiert allerdings ausgesprochen allergisch auf Kennungen, die nicht zwischen 0 und 15 liegen. Da keine Überprüfungen erfolgen, können unvorhersehbare Zugriffe erfolgen (durch Index-Überlauf in 'p_drvx'). Natürlich gibt es auch Schwierigkeiten beim Abfragen des Laufwerk-Bits in 'drvmap'.

Die GEMDOS-Programmierer waren sicherlich der Ansicht, daß es in Anwenderprogrammen keine Programmierfehler gibt, die illegale Kennungen erzeugen. Wie üblich muß der Anwender selbst nach den Bombenlegern im GEMDOS suchen. Bei Accessories ist wieder einmal zu beachten, daß diese unter dem PD der Hauptapplikation, also unter AES/Desktop, laufen, wodurch sich Konflikte ergeben können.

Funktion $19 Degetdrv

int Dgetdrv()

Hiermit erhält man die Kennung des für den aktiven Prozeß gültigen aktuellen Laufwerks. Dies kann entweder die zuletzt mit 'Dsetdrv' gesetzte oder die vom Parent-Prozeß "geerbte" Kennung sein.

Arbeitsweise

'Dgetdrv' ist ein Einzeiler. Es erfolgt direkt die Rückgabe von 'p_defdrv' aus dem PD.

Funktion $36 Dfree

*void Dfree (DISKINFO *info, int drv)*

Diese Funktion gibt Auskunft über den freien Speicherplatz auf dem Laufwerk 'drv' (0für aktuelles Laufwerk; 1,2,... für A:,B:,..). 'info' muß dabei auf freien Speicherplatz zeigen, der von 'Dfree' mit einer DISKINFO-Struktur (Abb. 4) gefüllt wird. Daraus kann der belegte und noch freie Speicherplatz in Bytes berechnet werden:

free = b_free * b_clsiz * b_secsiz

used =(b_total - b_free)*b_clsiz *b_secsiz

```
 1:  /* Routinen zum Abmelden von Laufwerken
 2:     bzw. zur Freigabe internen GEMDOS-Speichers
 3:
 4:     by A. Esser, Juni 1987
 5:     entwickelt mit MEGAMAX C
 6:  */
 7:
 8:  #define pokeb(a,b)   poke(0,(long)a,(int)b)
 9:  #define pokew(a,b)   poke(1,(long)a,(int)b)
10:  #define pokel(a,b)   poke(3,(long)a,(long)b)
11:  #define peekb(a)     ((int)peek(0,(long)a))
12:  #define peekw(a)     ((int)peek(1,(long)a))
13:  #define peekl(a)     peek(3,(long)a)
14:
15:
16:  /* allgemeine Funktionen zum Speicherzugriff
17:     ermöglicht Zugriff auf geschützte Bereiche
     und auf ungerade Adressen */
18:
19:  asm
20:  {
21:  poke:
22:     pea      spoke
23:     move.w   #38,-(A7)      ;XBIOS-Supexec
24:     trap     #14
25:     addq.w   #6,A7
26:     rts
27:  spoke:
28:     move.l   14(A7),A0      ;Poke-Adresse
29:     lea      18(A7),A1      ;Adresse des zu
                                 schreibenden Wertes
30:     move.w   12(A7),D1      ;Zahl der Bytes -1
                                 (0,1,3)
31:     bne.s    spokel         ;-> int oder long
32:     addq.w   #1,A1          ;char wurde als int
                                 übergeben
33:  spokel:
34:     move.b   (A1)+,(A0)+
35:     dbf      D1,spokel
36:     rts
37:  ;
38:  peek:
39:     pea      speek
40:     move.w   #38,-(A7)      ;XBIOS-Supexec
41:     trap     #14
42:     addq.l   #6,A7
43:     rts
44:  speek:
45:     move.l   14(A7),A0      ;Peek-Adresse
46:     moveq    #0,D0
47:     move.w   12(A7),D1      ;Zahl der Bytes -1
                                 (0,1,3)
48:  speekl:
49:     asl.l    #8,D0
50:     move.b   (A0)+,D0
51:     dbf      D1,speekl
52:     rts
53:  }
54:
55:
56:  /* gesamten internen Speicher eines Laufwerks
        freigeben */
```

geht weiter...

Dabei ist zu berücksichtigen, daß 'Dfree' nur die Informationen der FAT berücksichtigt, wo der Speicherplatz nach Clustern verbucht wird.

Arbeitsweise

Nach der Ermittlung des gewünschten Laufwerks (Berücksichtigung des aktuellen Laufwerks) und der Überprüfung mit 'd_chkdrv' werden alle FAT-Einträge (von 2 bis 'numcl') gelesen. Alle Einträge gleich Null werden als frei gezählt (dies ergibt 'b_free').

Dann wird die DISKINFO-Struktur erzeugt. Die Werte 'b_total', 'b_secsiz' und 'b_clsiz' entnimmt man dabei dem DMD ('d_numcl', 'd_recsiz', 'd_clsiz').

Für das Lesen der FAT wurde keine große, programmtechnische Mühe "vergeudet". Dies erledigt nämlich die allgemeine FAT-Lese-Routine, die mit 'f_seek' (der internen Version von GEMDOS-'Fseek') und 'f_read' jeden Eintrag einzeln liest und auch noch gleich den Folge-Cluster bestimmt, was hier ja überflüssig ist. Dieser enorme Overhead erklärt, warum 'Dfree' so berüchtigt langsam ist (gar nicht zu reden vom DESKTOP-"Disk-Info", das auch noch sämtliche Directories liest!). Eine auf 'Dfree' zugeschnittene Routine, die effektiver arbeitet, würde hier schon Wunder wirken.

Vorausschau

Das nächste Mal kehren wir wieder zur Dateiverwaltung zurück. Dabei geht es um die nächsthöhere Ebene - die Ebene der File-Descriptoren.

Alex Esser

```
57:   void free_drv(drive)
58:   register int drive;  /* Laufwerks-Kennung
                                0..15 */
59:   {  long dummy[4];     /* Dfree-Puffer */
60:      long old_bpb;      /* Zwischenspeicher für
                               alte Vektoren */
61:     long old_rw;
62:     long old_med;
63:     extern free_bpb(),free_rw(),free_med();
64:
65:     old_bpb = peekl(hdv_bpb);    /* alte Vektoren
                                        merken */
66:        old_rw = peekl(hdv_rw);
67:        old_med = peekl(hdv_mediach);
68:        pokel(hdv_bpb,free_bpb);  /*auf Spezial-Rou-
                                        tinen setzen */

69:        pokel(hdv_rw,free_rw);
70:        pokel(hdv_mediach,free_med);
71:      Dfree(dummy,drive+1);       /* Disk-Zugriff
                                        gibt Speicher
                                        frei */

72:      pokel(hdv_bpb,old_bpb);     /* Vektoren wieder
                                        zurückholen */
                                                    -

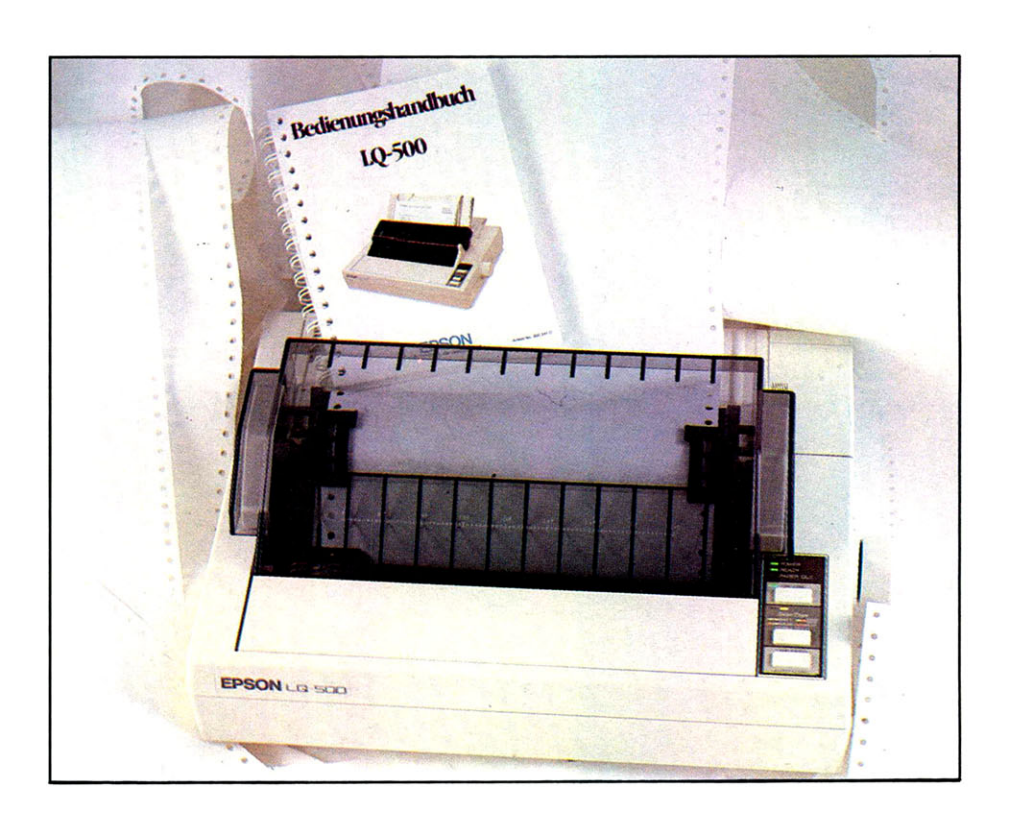
73:        pokel(hdv_rw,old_rw);
74:        pokel(hdv_mediach,old_med);
75:      return;
76:
77:          /* folgende hdv-Routinen sorgen indirekt
                für Speicher-Freigabe */
78:       asm
79:       {  free_med:
80:              moveq #2,D0          ;"sicherer
                                         Medium-Wechsel"
81:              rts
82:          free_rw:
83:              moveq #-14,D0        ;"Medium-Wechsel"
84:              rts
85:          free_bpb:
86:              moveq #0,D0          ;Medium nicht
                                       vorhanden
87:              rts
88:       }
89:   }
90:
91:
92:          /* internen Speicher aller Laufwerke
                freigeben */
93:   void free_all()
94:   {  register int drive;
95:
96:      /* ohne Rücksicht auf Existenz */
97:      for (drive = 0; drive < 16; drive++)
98:          free_drv(drive);
99:   }
```

ENDE



EPSON LQ-500

EPSON LQ-500 Die Konkurrenz auf dem Markt der 24 Nadeln ist groß. NEC stößt mit seinem neuen P 2200 in Preisbereiche vor, die noch vor Jahresfrist für solche Drucker reines Wunschdenken waren. Die Firma Epson, sonst nicht als Low-Price-Anbieter bekannt, hält mit ihrem neuesten Produkt, dem LQ-500 dagegen.



Nach der feierlichen Zeremonie des Karton-Öffnens findet der erstaunte Kunde, über den Fußboden verteilt, diverse Klappen, einen Stacheltraktor und ein deutsches Handbuch. Der Blick ins Innere des Druckers zeigt sauber verarbeitete Mechanik, in der sich hier und da sogar noch ein Metallteil erspähen läßt. Wurde gleich ein automatischer Einzelblatteinzug mitgeordert, erhöht sich die Klappenanzahl um zwei, zusätzlich zum Einzug selbst. Ich hatte im Laufe des Testes ständig das Problem der Unterbringung, denn viel Plastik will auch verstaut werden.

Die Menge der Einzelteile ergibt sich aus der Konzeption des LQ-500. Der Stacheltraktor ist nicht fest im Drucker eingebaut, sondern muß beim Wechsel auf Endlospapier montiert werden. Er erfordert eine andere Abdeckung des Druckbereiches als die Einzelblattführung und als der automatische Einzug.

Nachdem also erstmal getüftelt werden muß, was wohin gehört und welches Teil wo einrasten muß (das ausführliche und reich illustrierte Handbuch ist hier eine große Hilfe), geht es beim zweiten Mal dann schon schneller. Mit einiger Routine ist beim häufigen Wechsel der Papiersorten die Montage zwar einfach, aber doch lästig.

Exzellente Schrift

Hat der Drucker seinen Platz in der Nähe des Rechners gefunden und ist er via Parallelport mit ihm verbunden, muß natürlich erst noch die Farbbandkassette eingesetzt werden. Dabei ergeben sich keine Probleme, sogar die Hände bleiben sauber. Dann geht es an den ersten Druck. Hier zeigt der LQ-500 seine Stärken: Beide Zeichensätze, die er serienmäßig bietet, sind von hervorragender Qualität. Auch die Druckgeschwindigkeit ist gut, das dabei entwickelte Geräusch kann man als normal bis laut bezeichnen. Beim Druck größerer Texte oder Graphiken kommt er nicht ins Schwitzen, obwohl er über keinen Lüfter verfügt. Wer den Wunsch nach noch mehr Schriftvielfalt hegt, kann weitere Zeichensätze in Form steckbarer Module nachkaufen.

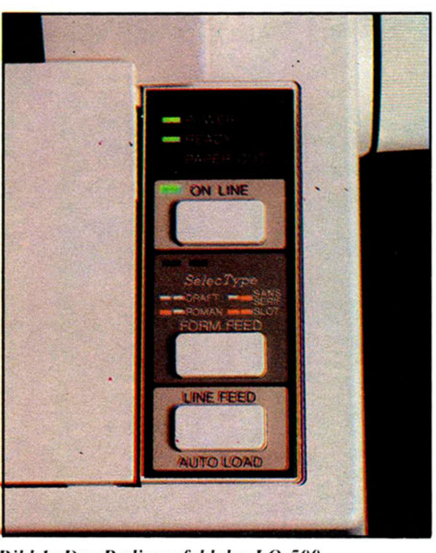


Bild 1: Das Bedienerfeld des LQ-500

Die Entwurfsqualität ist wie bei allen 24-Nadlern etwas dünn, was auf den geringeren Nadeldurchmesser zurückzuführen ist. Bei der Geschwindigkeit, die der LQ-500 in Briefqualität bietet, ist aber der Einsatz von Draft ohnehin selten nötig.

Steuerung von Hand

Wie bei jedem Drucker können die wichtigsten Einstellungen am LQ-500 über ein Bedienungsfeld vorgenommen werden. Der erste Blick enttäuscht zunächst. Während andere Hersteller mit bunt leuchtenden, an Raumschiffcockpits erinnernden Tastenfeldern aufwarten, kommt der Epson mit drei grauen Schaltern daher (siehe Bild 1). Mit ihnen wählt man On/Off Line, die Schriftart und schiebt das Papier um eine Zeile weiter. Ist der Drucker Off Line, wird mit einer ein Seitenvorschub ausgelöst, mit der zweiten ein Einzelblatt aus der dafür vorgesehenen Führung eingezogen.

Ich vermisse hier die Möglickeiten, die Schriftbreite per Taste zu ändern und den Druckerspeicher zu löschen. Letzteres ist nur mit dem Netzschalter zu bewerkstelligen und der sitzt, wie leider üblich, verrenkungsanfällig links hinten am Gehäuse.

Die DIP-Schalter sind von oben leicht zugänglich unter einer Klappe angeordnet. Hier befindet sich auch der Steckplatz für ein Zeichensatzmodul. Wer, wie ich, häufig von automatischem Einzelblatt zu Endlospapier wechselt, ist Epson dankbar für die Bedienerfreundlichkeit. Denn dabei muß jeweils nur ein DIP-Schalter bewegt werden. Insgesamt sind es derer 16, die außer der Wahl des internationalen Zeichensatzes auch die Einstellung der Vorgabewerte für Seitenlänge und z.B. das Überspringen der Perforation bei Endlospapier ermöglichen.

Der LQ-500 läßt sich auf drei Arten mit Papier versorgen: Endlospapier wird mit einem Zugtraktor transportiert, Einzelblätter sind mittels einer Führung und 'Autoload' oder mit automatischem Einzug einzuziehen. Bei allen drei Arbeitsweisen hatte ich keine Probleme mit der Führung des Papiers. Es saß immer gerade und glatt im Drucker, jedoch ist der LQ-500 nicht in der Lage, Papier zurückzufahren. Jeder verzagte Griff zur Transportwalze wird mit Chaos im Druckwerk beantwortet. Der automatische Einzug hält bis zu 100 Blatt in Vorrat, die er auch sicher ablegt. Es ist die Verarbeitung von bis zu DIN A4 breitem Papier erlaubt, eine A3-Version gibt es nicht. Laut Handbuch sind bis zu 2 Durchschläge, allerdings nur im Endlosmodus, möglich.

Besonders bei Briefen ist es wichtig, daß der Drucker im Einzelblattmodus oben und unten möglichst wenig Rand fordert. Der LQ-500 läßt hier nur jeweils zwei Zeilen frei, auf einem DIN A4- Blatt bleiben dann immerhin noch 66 Zeilen. Zum Vergleich: beim Star NB-24 sind es beispielsweise nur 59.

Die Druckersoftware

Was kann der LQ-500 nun außer drei Zeichensätzen? Die Abbildungen zeigen einige Optionen. Bei den verschiedenen Stilen fallen besonders Umriß- und Schattendruck auf. Schon selbstverständlich für einen Drucker dieser Art ist Proportionalschrift, die aber nur in Pica-Breite nutzbar ist. Eine weitere Option ist für Freunde eines PC-Emulators wichtig: Die Zeichen ASCII 128-255 lassen sich zwischen kursivem und IBM-Graphikzeichensatz umschalten. Das kann sowohl mittels DIP-Schalter als auch über Software geschehen. Leider kann die 'Autoload'-Funktion, das Einziehen eines Einzelblattes aus der Führung, nicht mit dem Rechner gesteuert werden, wohingegen der automatische Einzelblatteinzug programmierbar ist.

LQ-500 und ST

Die schönsten Druckoptionen nutzen herzlich wenig, wenn bestehende Software sie nicht unterstützt. Kom-

```
Der LQ-500 von Epson besitzt zwei
eingebaute Letter-Quality Zeichensätze:
Das ist Roman. Das ist Sans Serif.
Und natürlich Entwurfsqualität, die
sich durch die Geschwindigkeit von
180 Zeichen pro Sekunde auszeichnet.
Der LQ-500 beherrscht besondere Effekte:
Den Umrißdruck, auch Outlined genannt,
und Schattendruck.
In Elite-Schmaldruck sind 160 Zeichen pro Zeile möglich.
```

Bild 2: Die verschiedenen Zeichensätze des LQ-500

```
Diese Zeile in fett, unterstrichen, kursiv, super, sub.
Diese Zeile in fett, unterstrichen, kursiv, super, sub.
Diese Zeile in fett, unterstrichen, kursiv, super, sub.
Diese Zeile in fett, unterstrichen,
kursiv, super, sub.

Der Atari-Zeichensatz ohne hebräische Zeichen:
  ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
@ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [ \ ] ^ _
` a b c d e f g h i j k l m n o p q r s t u v w x y z { | } ~ △
Ç ü é â ä à å ç ê ë è ï î ì Ä Å É æ Æ ô ö ò û ù ÿ Ö Ü ¢ £ ¥ ß ƒ
á í ó ú ñ Ñ ª º ¿ ⌐ ¬ ½ ¼ ¡ « » ã õ Ø ø œ Œ À Ã Õ ¨ ´ † ¶ © ® ™
                                                          § ∧ ∞
∞ ß Γ π Σ σ µ τ Φ θ Ω δ ∮ ø ∈ ∩ ≡ ± ≥ ≤ ⌠ ⌡ ÷ ≈ ° • · √ ⁿ ² ³
```

Bild 3: Arbeiten mit 1st-Word

```
Dieser Text wurde mit
Wordplus erstellt und
unter 1ST Proportional
gedruckt.
Aufgrund seiner Kompa-
tibilität zu den gros-
sen Brüdern LQ-800 und
LQ-1000 gelingt der
Druck mit dem LQ-500
sofort mit den dafür
vorgesehenen Treibern.
```

Bild 4: Auch Proportionalschrift ist mit Softwareunterstützung möglich

patibilität heißt das Zauberwort. Da Epson im Bereich der Steuercodes selbst einen Standard gesetzt hat, besteht bei Textverarbeitungssoftware kein Problem. Aufgrund seiner Kompatibilität mit den großen Brüdern LQ-800 und LQ-1000 können für den LQ-500 die bestehenden Druckeranpassungen genutzt werden. Ausdrucke mit Wordplus und 1st Proportional gelangen sofort, wenn man mal von den Sonderzeichen absieht. Hier mußte der mitgelieferte Treiber für Wordplus modifiziert werden (siehe Bild 3). Die hellen Zeichen ergeben sich dadurch, daß es diese partout nicht im Zeichensatz des Druckers gibt und sie, mehr schlecht als recht, durch 8-Nadel Graphik nachgebildet werden müssen.

Ein weites Einsatzgebiet für einen Drucker am ST ist die Graphik. Wenn überhaupt, so unterstützt Graphiksoftware von den 24-Nadlern oft nur den NEC P6. Dieser Drucker verfügt jedoch über eine maximale Auflösung von 360 x 360 Punkten pro Zoll. Beim Epson sind es senkrecht nur 180. Daraus ergeben sich hier und da Probleme. Z.B. nutzt CADproject diese Auflösung nicht, daher kann nur im 180 x 180 Punkte-Modus gedruckt werden. Nach geringer Modifizierung des Treibers für den NEC P6 gelang auch der Druck mit Stad. Bei Signum kann ja ohnehin die Auflösung 360 x 180 gewählt werden, so daß es auch hier keine Schwierigkeiten gibt.

Ein weiteres Problem für jeden Besitzer eines 24-Nadel Druckers sind die Hardcopies. Viele Malprogramme lassen nur diese Art des Ausdrucks zu. Doch der ST unterstützt dabei bekanntlich nur 9 Nadeln. Es gibt aber im PD-Service der Redaktion zwei Programme, die hier Abhilfe schaffen. Das ist zum einen das Hardcopy-Accessory für den LQ-800, zum anderen der FX-Emulator der Firma NEC, der 11 verschiedene Hardcopy-Modi anbietet. Mit beiden Programmen gelingen Hardcopies per ALT-Help in teilweise bestechender Qualität.

Fazit

Der LQ-500 macht rundherum einen ordentlichen Eindruck. Am ST wird er von der wichtigsten Software unterstützt. Die Bedienung ist einfach, das Handbuch gibt ausführlich Auskunft; wenn man mal von einigen Druckfehlern absieht, ist es gelungen. Das Bedienungsfeld ist leider etwas mager ausgefallen und die Prozedur des Traktor-Auf- und Abbaus ist keine Freude. Außerdem braucht man für die vielen Teile ein zusätzliches Fach im Regal. Epson gewährt auf das Gerät selbst eine einjährige Garantie, die den Druckkopf jedoch nicht mit einschließt. Trotzdem machen seine robuste Verarbeitung, seine gute Schriftqualität und nicht zuletzt der niedrige Preis den LQ-500 zu einem empfehlenswerten Gerät.

Ingo Brümmer

Epson LQ-500
Technische Daten:
***Schriftbreiten**: 0,12,15 CPI*
***Zeichensätze**: 13 Internationale, Kursiv/IBM-Graphik 2 Letter Quality, Draft, Fontmodule*
***Speicher**: /8kByte wählbar*
***Auflösung horizontal**: 60, 80, 90, 120, 180, 240, 360 Punkte/Zoll*
***vertikal**: maximal 180 Punkte/Zoll*
Geschwindigkeit: LQ - 50/60 Zeichen/s (10/12 CPI)
Draft - 150/180 Zeichen/s (10/12 CPI)
Kompatibel zu Epson LQ-800,LQ-1000
***Lebensdauer**:*
des Druckkopfes: 200 Millionen Zeichen/Nadel
des Farbbandes: 2 Millionen Zeichen bei LQ
***Preise**: Drucker inkl. Stacheltraktor und Einzelblattzuführung: 1098 DM*
Druckkopf: 275 DM
Automatischer Einzelblatteinzug: 225 DM
Schriftmodule: 80 DM
Farbband: 28 DM
+ Gutes Preis-/Leistungsverhältnis
+ Gute Schriftqualität
+ 2 Schriftarten serienmäßig
+ DIP-Schalter gut zugänglich
- Komplizierter Papierwechsel
- Druckkopf ohne Garantie

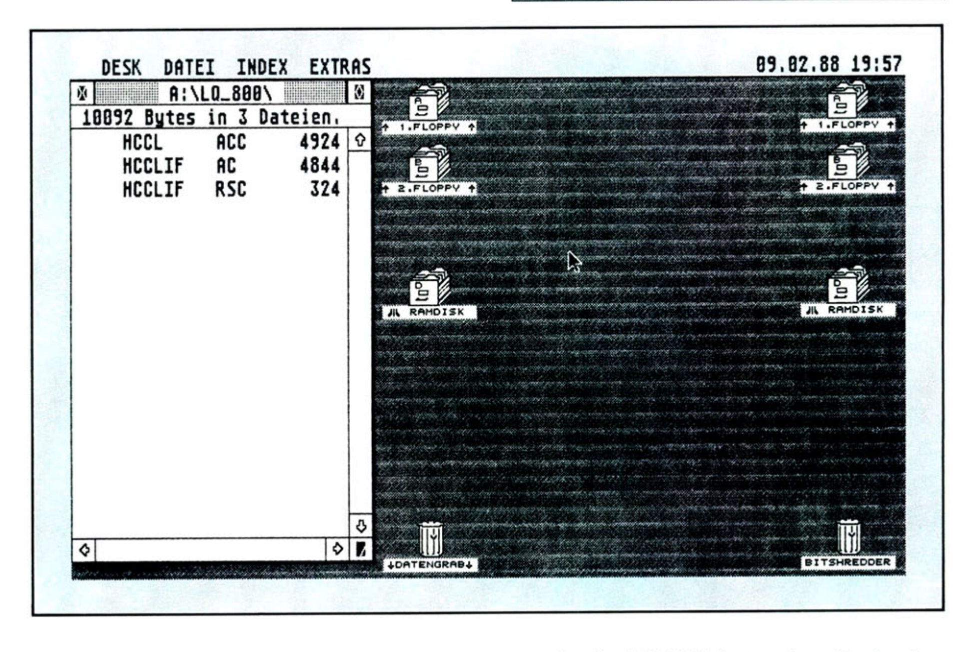


Bild 5: Das im PD-Service erhältliche Hardcopy-Accessory für den LQ-800 bringt saubere Hardcopies.

ENDE

TEST

PREISBRECHER FÜR RISIKOFREUDIGE

Jeffersons Modula

Für den Atari gab es bisher nur den TDI Modula-2-Compiler. Jetzt erreichte uns ein Konkurrent aus den USA: Jefferson Modula-2. Kann das System dem Referenzcompiler aus Großbritannien Paroli bieten, und bekommt man für unter DM 100,- ein Billigprodukt oder einen Preisbrecher ?

Geliefert wird das Entwicklungspaket, für das es noch keinen deutschen Distributor gibt, auf zwei einseitigen Disketten und einem kleinen Manual. Im Test steht hier die Version 1.1 in der "kleinen" Version, die Sie für $49.95 in den Staaten erwerben können. Bei einem momentanen Dollarkurs von DM 1,70 also ein Preisbrecher für ca. DM 85,-, zu denen allerdings noch etwa DM 15,- bis 20,- an Steuern und Zoll kommen.

```
yet another shell for  Atari ST

(non-profit redistribution is permitted)

»guläm«  Copyright © 1987 pm@Case

61 built-in commands:
alias    dirc      exit     history msoff    pwd      source   unalias
cd       dirs      fg       if      mv       rehash   sx       unset
chmod    echo      fgrep    lpr     peekw    ren      te       unsetenv
copy     egrep     foreach  ls      pokew    rm       teexit   which
cp       ef        format   mem     popd     rmdir    time     while
date     endfor    gem      mkdir   print    rx       touch
df       endif     grep     more    printenv set      ue
dm       endwhile  help     mson    pushd    setenv   uekb

programs accessible through hash table:
oops   m2link m2dec  m2
d:\jm2 58#
d:\jm2 58# ls bin
bin

d:\jm2\bin:
m2.ttp     m2dec.ttp  m2link.ttp oops.tos
d:\jm2 59#
```

Bild 1: Jefferson Modula-2 bietet über 60 Kommandos

Installation

Bei der Installation geht Jefferson Software einen bisher seltenen Weg: Alle Dateien sind mit dem Kompressionsprogramm ARC platzsparend auf den Disketten untergebracht. Die mitgelieferte Public-Domain-Shell Guläm (dazu noch mehr weiter unten) übernimmt per Batchfile das Kopieren auf Ihre Disketten oder Festplatte. Um das System in einem bestimmten Ordner zu plazieren, sind einige Zeilen mit Zugriffspfaden in dem Batchfile zu ändern. Nach ca. 20 Minuten (alle Zeiten mit Festplatte gemessen) stehen 760 KB Programme und Libraries auf dem Zielspeicher. Einen Kopierschutz gibt es glücklicherweise nicht.

Arbeitsumgebung

Im Gegensatz zum TDI-Compiler benutzt Jefferson (JS) Modula keine GEM-Funktionen. Compiler und Linker sind TTP-Programme und schreiben ihre Meldungen auf die Standardausgabe. Das ist durchaus vertretbar, denn die Textfenster bei TDI bringen keinen wirklichen Gewinn.

Dennoch gibt es eine besondere Umgebung: die Public-Domain-Shell Guläm in der Version 05. Auch wenn hier die Früchte fremder Arbeit geerntet werden, wird das System erst durch diese Shell zu einem gut benutzbaren Entwicklungspaket.

Guläm ist m.E. die leistungsfähigste Public-Domain-Shell. Sie besitzt eine hervorragende Alias-Funktion, mit Setenv können Suchpfade bei der Programmausführung gesetzt werden, Batchfiles arbeiten mit Schleifen, Bedingungen und Parameter. Dateinamen kann man bei der Eingabe mit der Insert-Taste von Guläm vervollständigen lassen. Insgesamt gibt es über 60 Kommandos, die alles Notwendige und ein bißchen mehr bieten (Bild 1). Bei voller Ausnutzung des Alias-Befehls kann man sich eine Umgebung schaffen, die genau den persönlichen Wünschen entspricht. Leider ist die Shell auf eine amerikanische Tastatur eingestellt, läßt sich allerdings für deutsche Verhältnisse patchen.

Einen Editor benötigt JS-Modula nicht, denn der bekannte und gute EMACS ist in einer etwas abgemagerten Version im Guläm enthalten.

Ladezeiten entfallen hier also. Man kann aber auch einen GEM-Editor aufrufen, da Guläm mit dem Präfix "gem" die TOS-Ebene verläßt.
Allerdings hat dieses Vorgehen ohne speziellen Editor natürlich den entscheidenden Nachteil, daß Compilerfehler nicht wie bei TDI automatisch im Quelltext angezeigt werden. Vielmehr muß man zunächst das Hilfsprogramm OOPS laufen lassen, das aus den Informationen der Fehlerdateien einen Klartext erzeugt, in dem alle fehlerhaften Quelltext-Zeilen samt Meldungen enthalten sind. Bei der Fehlerkorrektur muß man dann diesen Text einblenden, was mit EMACS kein Problem ist. Das Zusammenspiel zwischen dem Quelltext und den Fehlermeldungen ist bei TDI allerdings erheblich besser gelöst.

```
d:\jm2 64# m2 allqueen.mod
Jefferson Software Modula-2 Compiler V-1.00

allqueen.mod
 - d:\jm2\std\InOut.SBM
 + AllQueens.RFM
 + AllQueens.OBM 584
d:\jm2 65#
d:\jm2 65# m2link allqueen
Jefferson Software Linker V-1.00

 - 0000003C allqueen.OBM
 - 000002EE d:\jm2\std\InOut.OBM
 - 00001246 d:\jm2\std\FileSystem.OBM
 - 00001EE8 d:\jm2\std\System.OBM
 - 000028DC d:\jm2\tos\GEMDOSFiles.OBM
 - 00002D26 d:\jm2\std\FPSys.OBM
 + 00003562 allqueen.TOS
d:\jm2 66# ls -l allqueen.*
-rw------- 1 u          783 Feb 13 19:34 allqueen.mod
-rw------- 1 u          720 Feb 18 15:38 allqueen.obm
-rw------- 1 u          357 Feb 18 15:38 allqueen.rfm
-rw------- 1 u        13881 Feb 18 15:39 allqueen.tos
d:\jm2 67#
```

Bild 2: Der Compiler wird einfach mit dem Namen des Quellprogramms aufgerufen.

Compiler & Linker

Der Compiler wird - wie in Bild 2 zu sehen - einfach mit dem Namen des Quellprogramms aufgerufen. Dabei sind noch die Option -R und -V zum Umschalten der Bereichs- und Overflowüberprüfung möglich. Weitere Compileroptionen gibt es leider nicht.
Während des Übersetzerlaufs gibt JS-Modula auf dem Bildschirm die Namen der importierten Module und erzeugten Dateien aus. Das System ist eine Implementierung nach der neuesten (dritten) Revision von Modula-2 (siehe Info-Kasten im Bericht über Megamax-Modula in diesem Heft). Daher kann der Compiler den Quelltext in nur einem Durchlauf übersetzen und müßte somit schneller als TDI sein. Zur Einschätzung der Geschwindigkeit habe ich drei Programme durchlaufen lassen (Bild 3). Dabei zeigt sich, daß der Compiler keineswegs optimal programmiert ist und bei größeren Programmierprojekten auf jeden Fall zu langsam wird.
Bei den Tests zeigten sich einige herbe Überraschungen: Das InOut-Modul ist fehlerhaft und die damit compilierten Programme führen zum Absturz. Dadurch wird das System leider nur sehr eingeschränkt nutzbar. Ich nehme an, daß es sich um einen Fehler bei der Zusammenstellung des Systems handelt. Weiterhin sind LONGREALs entgegen den Aussagen im Handbuch nicht vorhanden. Ebenso gibt es keine Coroutinen, womit das System nur eine eingeschränkte Implementierung des Modula-2-Sprachumfangs ist.
Jefferson Software bietet einen umfangreichen Support an. Auch wird man im Handbuch immer wieder aufgefordert, sich bei Problemen telephonisch oder per Mailbox zu melden. Wahrscheinlich wäre der Fehler in InOut auf diese Weise schnell behoben - für uns Europäer allerdings sehr kostspielig. Updates, über die man bei Registrierung automatisch Nachricht erhält, sind für $ 10 erhältlich.
Der Linker zeigt sich vom Aufruf und den Ausgaben her als dem Compiler ähnlich. Er arbeitet mit zufriedenstellender Geschwindigkeit, kann aber nicht so gut optimieren wie der TDI-Linker.
Die Länge der erzeugten Programme deutet darauf hin, daß der Linker alle Routinen eines Moduls einbindet, auch wenn sie nicht benutzt werden. Bei größeren Programmen kann sich nach meinen Erfahrungen dabei eine Verdoppelung der Programmlänge ergeben.
Bei der Verwendung von zu langen Pfadnamen für die Bibliotheken stürzt der Linker ab. Bei der Installation sollte man das System also möglichst weit oben im Dateibaum plazieren.

Die Bibliotheken

Die Library-Module werden größtenteils auch mit Source-Code ausgeliefert, was sehr erfreulich ist und bei

Modula-2 Compiler- und Linkerzeiten

Nr.	Compilezeit		Linkzeit		Programmgröße		Programm
	JS	TDI	JS	TDI	JS	TDI	
1	0:17	0:20	0:20	0:19	5164	5442	Backtracking
2	0:21	0:18	0:21	0:18	5051	2244	Rekursion
3	0:38	0:34	0:43	0:52	17537	14332	größere IMPORTs

Bild 3: Modula-2 Compiler- und Linkerzeiten

LESEN SIE SCHEIBENKLEISTER!



SCHEIBENKLEISTER - MASSENSPEICHER AM ST.
Alles über Floppies, Festplatten usw. von Claus Brod und Anton Stepper.
Was steht drin?



Kursteil (für die ganze Familie):
- Floppyprogrammierung mit allen erlaubten und unerlaubten Mitteln (per BIOS, XBIOS, GEMDOS und direkter Controllerprogrammierung);
Kopierschutz, Aufzeichnungsverfahren, Datenstrukturen auf der Diskette
- Hardwaredokumentation zu Floppy und Festplatte (Anschluß von Fremdlaufwerken, Justierung, Reparaturhinweise)
- Festplatte: Prinzip, Controller, Programmierung

Nachschlageteil (für Programmierer):
- Hard- und Softwarereferenz zu DMA-Chip, Floppycontroller, Festplattencontroller
- GEMDOS-, BIOS- und XBIOS-Funktionen zur Massenspeicherprogrammierung (auch als GFA-BASIC-Bibliothek auf Diskette)

Software (für alle, fertig zum Anwenden mit kompletten Anleitungen) :
- TED, der Trackeditor: Formate analysieren, ändern, erstellen
- SED, der RAM-Disk/EPROM-Disk/Floppydisk/Harddisk-Monitor: Ordnernamen ändern, gelöschte Dateien retten, spezieller Harddiskmonitor für direkten Festplattenzugriff (eigene Formatierroutine für zwei MB mehr)
- neue HYPERFORMAT-Version 3.0: MS-DOS -kompatibles Format, bis zu 950 KB auf doppelseitiger Diskette, superfixe Formatierroutinen (optional unter 20 Sekunden für doppelseitige Disketten), Schnelladeformate
- Steprateneinstellung, Konvertierung von Disketten auf Schnelladeformat
- Assemblerroutinen für direkten Floppy- und Festplattenzugriff zum Einbinden in eigene Programme
- lauffähige Programme mit Quelltext auf Diskette

Ca. 600 Seiten, Buch mit Diskette für 59DM, erscheint Ende Februar

---

HIERMIT BESTELLE ICH___EXEMPLARE VON "SCHEIBENKLEISTER, MASSENSPEICHER AM ST".
MIT DISKETTE FÜR DM 59.--
ANRUF GENÜGT: 06196/481811.
MO-FR 9-13 UND 14-17 UHR
SCHRIFTLICHE BESTELLUNG NUR GEGEN

VORKASSE ODER NACHNAHME (VERSANDKOSTEN DM 5.50: BEI NACHNAHME ZUZÜGLICH DM 4.70)
NAME:________
VORNAME:________
STRASSE:________
ORT:________
UNTERSCHR. :________



MERLIN COMPUTER GMBH
INDUSTRIESTRAßE 26
6236 ESCHBORN
TEL. 06196/481811

TDI in der Commercial-Version teuer zu erkaufen ist. Die Standardmodule sind leider nicht ganz vollständig und teilweise fehlerhaft.
Die systemspezifischen Module für GEM, GEMDOS und BIOS sind etwas zu sehr aufgesplittet, trotzdem aber vollständig und brauchbar. Module für eine automatische Fensterverwaltung oder andere Komfort-Bibliotheken sind leider nicht vorhanden - TDI verkauft sie extra im Toolkit.

Benchmarks

Letztlich entscheidend über die Qualität eines Compilers ist die Effizienz des erzeugten Codes. Dabei steht für die meisten die Geschwindigkeit im Vordergrund. Allerdings ist es sehr schwierig, ein Produkt mit Benchmarks richtig einzuschätzen, da die verwendeten Programme oftmals einseitig testen. Ich habe daher eine ganze Reihe von Geschwindigkeitstests durchgeführt, die auch für zukünftige Modula-2-Besprechungen verwendet werden sollen.
Die Benchmarks umfassen 30 Miniprogramme, die sich jeweils auf einen Teilaspekt konzentrieren. Es gibt Programme, die die Geschwindigkeit der Schleifenkonstrukte testen, die Verarbeitung von Ausdrücken überprüfen und die herauszufinden versuchen, ob der Compiler intelligent optimiert. Gerade durch geschickte Optimierung kann das Compilat Geschwindigkeitssteigerungen erreichen. Denkbar sind Umstellungen der Ausdrücke, Speicherung von Zwischenergebnissen in Registern oder die Errechnung von konstanten Ausdrücken während der Compilierung. Die Ergebnisse dieser Tests sehen Sie in Bild 4. Allerdings muß hierbei berücksichtigt werden, daß JS-Modula keine Coroutinen kennt, die im allgemeinen einen kleinen Mehraufwand für das Laufzeitsystem bedeuten.
JS-Modula zeigt sich flink bei Schleifenkonstrukten und Parameterübergaben. REPEAT-Schleifen werden anscheinend hervorragend übersetzt. Auch die Auswertung von Expressions geht recht schnell.
Ansonsten ist JS-Modula dem TDI-Compiler ebenbürtig - wenn man von den Fließkommaoperationen absieht. Die sehr schlechte Verarbeitung von einfachen REAL-Operationen verstärkt sich bei der Bibliothek mit höheren Funktionen.
Die erzeugten Programme sind also annehmbar schnell. Die langsamen REAL-Operationen machen den Compiler allerdings für rechenintensive Anwendungen ungeeignet.

Modula-2 Benchmarks

Nr.	TDI Modula-2		JS Modula-2		testet ...
	Min.	Ticks	Min.	Ticks	
1	0:07	1334	0:07	1489	Prozeduraufruf
2	1:42	20339	1:33	18548	Addition
3	1:21	16277	1:18	15518	Increment
4	1:47	21418	1:38	19634	Additionsoptimierung
5	1:27	17311	1:23	16550	Increment als Vergleich
6	2:09	25835	1:57	23480	INTEGER-Addition
7	2:09	25833	1:57	23471	CARDINAL-Addition
8	1:11	14228	1:18	15543	FOR-Schleife
9	1:21	16278	1:02	12437	REPEAT-Schleife
10	1:21	16191	1:18	1552	WHILE-Schleife
11	1:04	12743	0:54	10791	INTEGER-Parameter
12	1:04	12739	0:54	10791	INTEGER VAR-Parameter
13	1:06	13130	0:59	11821	RECORD-Parameter
14	0:34	6755	0:30	5977	RECORD VAR-Parameter
15	0:49	9837	0:49	9780	Konstanten-Optimierung
16	0:51	10249	0:51	10201	Konstanten-Optimierung
17	1:28	17655	1:26	17171	Expression-Optimierung
18	1:42	16822	1:22	16354	Expression-Optimierung
19	0:37	7431	0:36	7102	Zwischenergebnis-Optimierung
20	0:37	7435	0:35	7099	Zwischenergebnis-Optimierung
21	0:09	1831	0:11	2154	IF-Statement
22	0:13	2645	0:13	2688	IF durch CASE ausgedrückt
23	0:38	7689	0:33	6667	CASE-Statement
24	0:40	7960	0:39	7804	CASE durch IF ausgedrückt
25	0:47	9441	1:03	12636	REAL-Arithmetik
26	2:05	24924	—	—	LONGREAL-Arithmetik
27	1:52	22380	5:42	68358	REAL-Library
28	1:21	16264	1:21	16135	String-Library
29	2:10	26075	2:07	25437	ARRAY-Zugriffe
30	0:09	1896	0:10	2056	RECORD-Zugriffe

Alle Zeiten mit time-Kommando von Guläm gemessen

Bild 4: Benchmarks

Handbuch

Die vorliegende Version kommt mit einem kleinen Handbüchlein mit 26 Seiten. In ihm werden nur die notwendigsten Dinge zu Installation und den Compiler- und Linkeraufrufen beschrieben. Für $ 80 erhält man eine "enhanced version", die ein ausführlicheres Handbuch mit Listings aller Bibliotheksmodule miteinschließt. Man kann mit dem kleinen Manual arbeiten, sollte sich dann aber alle mitgelieferten Sourcen ausdrucken.

Fazit

Die Installation ist hervorragend vorbereitet, läuft vollautomatisch und ist problemlos. Die Arbeitsumgebung ist Geschmacksache. Wer GEM-abhängig ist, wird sicherlich die TDI-Umgebung vorziehen. Es lohnt sich aber, die Möglichkeiten des leistungsstarken Gulām (nach Anpassung an die deutsche Tastatur) genauer anzuschauen, da schnellere Turn-Around-Zeiten zu erreichen sind.

Die Fehlerbehebung gestaltet sich umständlich und könnte besser unterstützt werden. Der Compiler selber ist noch zu langsam und eher eine Enttäuschung, auch wenn er seine Aufgabe vom erzeugten Code her gut bewältigt. Das Fehlen des Coroutinen-Konzepts ist ein echter Mangel, zudem es relativ leicht zu implementieren ist. Ebenso ist der Linker simpel implementiert und noch nicht optimal.

Viele kleinere und größere Fehler (InOut-Modul !) verursachen bei JS-Modula den Eindruck, daß bei der Zusammenstellung der Disketten sehr schlampig gearbeitet und nicht korrekt auf Bugs überprüft wurde. Das mag sich in den USA mit einem Telefonat schnell beheben lassen; ohne einen deutschen Vertrieb und Support hat man aber viele Schwierigkeiten. Die Risiken bei einem Kauf im Ausland zeigen sich hier deutlich. Für nur ein Drittel des Preises von TDI erhält man einen Modula-2-Compiler, mit dem man arbeiten kann. Bei sehr vielen Kleinigkeiten werde ich aber das Gefühl nicht los, es mit einem portierten Public-Domain-Compiler oder mit dem Ergebnis eines Universitätskurses über Compilerbau zu tun zu haben.

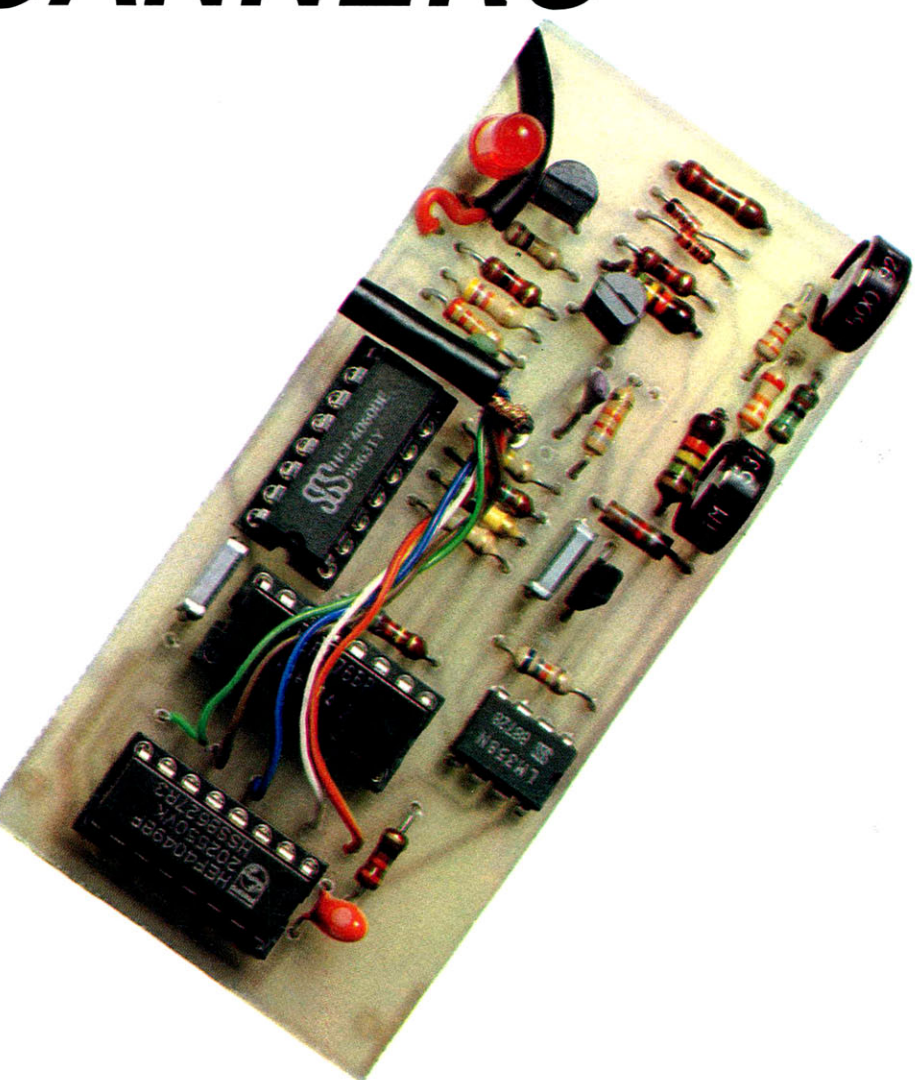
Bei der zu erwartenden Flut von Modula-Compilern für den ATARI (FTL, Megamax, ETH und andere Implementierungen) wird Jefferson Modula-2 vom Komfort her und in Feinheiten sicher seinen Meister finden, im Preis aber nicht. Trotzdem wäre eine gründliche Überarbeitung angesagt.

Robert Tolksdorf

ENDE

SELBSTBAU EINES COMPUTER-SCANNERS

Das Wort "Scanner" kommt aus dem Englischen. "To scan" bedeutet abtasten oder auch überfliegen. Der Hobby Computeranwender versteht darunter ein Gerät, das Bild-Vorlagen in (s)einen Computer einliest.



Im Gegensatz zu den schnellen Video-Digitizern wird beim Scanner eine lichtempfndliche Photozelle in kleinen Abständen über eine Vorlage geführt. Für jeden Abtastschritt werden dem Computer Helligkeitsdaten übermittelt. So wird die Vorlage Zeile um Zeile eingelesen.

Ausgehend davon kam ich - wie sicherlich andere schon vor mir - auf den Gedanken daß die Photozelle von meinem Drucker fortbewegt werden könnte. Dazu genügt es, sie in ein altes Farbbandgehäuse einzubauen.

Mit einer kleinen Sequenz von Steuerzeichen lassen sich die meisten Drucker zum Scannen zweckentfremden. Nach einigen hundert Horizontalschritten folgt ein Wagenrücklauf, um die nächste Zeile zu lesen. Die Vorlagen müssen sich hierbei alle in den Drucker einspannen lassen.

Wer einen Flachbettplotter sein eigen nennt, kann z.B. auch Plattenhüllen bequem abtasten. Ein Plotter macht außerdem keine Anstalten bei der Ansteuerung. Einige Drucker zeigen, dank modernster Wegeoptimierung, ein großes Maß an Starrsinn. Manche Geräte versalzen selbst einem trickreichen Programmierer die Suppe, indem sie stets einen Wagenrücklauf abwarten und dann emsig, aber viel zu schnell über die Vorlage flitzen.

Folgendes wird also zum Scannen benötigt:

1. Ein (einfacher) Drucker oder ein Plotter
2. Eine Photozelle mit logarithmischem A/D-Wandler
3. Scannersoftware mit USM-Algorithmen zum Verdoppeln der Bildschärfe
4. Scannenswerte Vorlagen

Die Sache sieht also nicht besonders schwierig aus, zumal meine komfortable Scannersoftware als Public Domain erhältlich ist.

Nur DM 25,- habe ich für meine Elektronik berappt. Nach 4-stündiger Nachtarbeit funktionierte der A/D-Wandler perfekt. Aber dann gab es doch Schwierigkeiten: Die ersten eingescannten Bilder wirkten unscharf und "verrauscht".

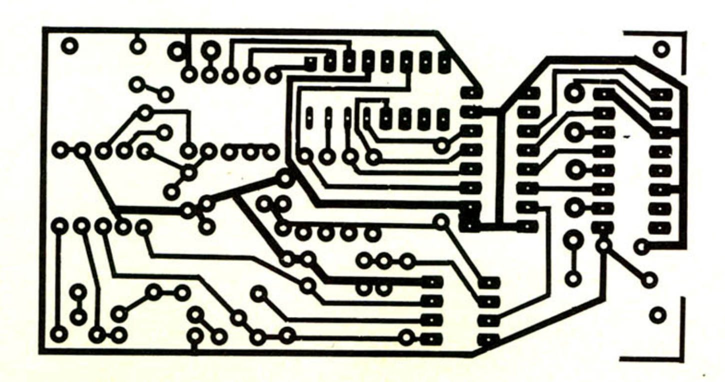
Erst nach und nach gelang, hauptsächlich durch verbesserte Software, ein Scanner mit erstaunlicher Bildqualität.

Die Hardware war, abgesehen von

einem kleinen Wackelkontakt, nicht weiter zu verbessern. Die umgewandelten Helligkeitswerte entsprechen, wegen der logarithmischen Kennlinie vom A/D-Wandler, dem tatsächlichen Graueindruck. Die Vorlagen werden außerdem nicht, wie bei vielen Billigscannern, mit Infrarotlicht abgetastet, sondern mit sichtbarem Licht.
Meine Scannersoftware bietet also einige Leckerbissen:
Das abgetastete Originalbild kann vergrößert, verkleinert und gestaucht werden. Auch Helligkeit, Kontrast und Gradation sind nachträglich zu verändern. So kann man z.B. die dunklen Bildbereiche verändern, ohne daß sich die hellen und mittelgrauen Töne verändern.
Die Grauabstufungen sind auf dem s/w-Monitor nur durch Füllmuster darstellbar. Diese Muster sind besonders fein, um ein möglichst realistisches Bild zu erzeugen. Ich habe auch darauf geachtet, daß keine "Kanten-Effekte" zwischen verschiedenen Mustern auftreten.
Die USM-Funktionen (USM = Unscharf-Maskierung) betonen feine Bilddetails, die beim Abtasten teilweise verlorengingen. Außerdem wird die "Treppchenbildung" vermieden. Diese Funktion kann tatsächlich den Schärfeeindruck verdoppeln. Das ist etwas, was andere Scannersoftware nicht bieten kann.
Das so aufbereitete Bild läßt sich im Degas- oder Binärformat auf Diskette abspeichern. Die Bilder können so mit einem Malprogramm noch weiter verschönert werden.
Mit einer geplanten Version 2.0 kann man in die Farbauflösung umwandeln und die Bilder sogar einfärben.

Hardware

Die Hardware besteht im wesentlichen aus einem Phototransistor, dem logarithmischen Verstärker und dem 4-Bit-A/D-Wandler.
Die Hardware wird mit dem Joystick-Port verbunden und die Elektronik über diesen Stecker auch mit 5 Volt versorgt. Man bekommt einen solchen Joystick-Stecker im Elektronik-Laden (9-polige D- Buchse). Man muß nur den Metallrahmen entfernen. Zur Not hilft auch das Kabel eines alten Joysticks.
Ich möchte nun ganz kurz die Funktionsweise meiner Schaltung erklären:
Der Phototransistor T3 (BPY 14) liefert einen zur Lichtstärke proportionalen Ausgangsstrom. T1 logarithmiert diesen Strom und wandelt ihn in eine Gleichspannung (maximal 0,7 Volt) um.
Widerstand R2 und C1 filtern Brummspannungen aus. Der nun folgende OP verstärkt diese Spannung. Hier sind die Regler für Abtasthelligkeit (P2) und Kontrast (P1) angebracht.
Transistor T2 steuert eine Leuchtdiode an. Diese LED leuchtet in Abhängigkeit von der "gesehenen" Helligkeit auf.
Alle übrigen ICs dienen zur A/D-Wandlung. Der Eingang befindet sich an IC 1 Pin 6. Beim Aufbau sollte man darauf achten, daß C4, R11 und R10 nicht zu weit weg von IC2 plaziert werden. Man sollte die Digital-Masse (breites Symbol) und Analog-Masse (schmales Symbol) getrennt verdrahten und erst an C6 miteinander verbinden.
Die gesamte Elektronik wurde auf einer Lochrasterplatine von 30 mm * 80 mm aufgebaut. Man tut sich bei der doppelten Platinenfläche allerdings weniger schwer. IC-Fassungen für IC2, IC3 und IC4 sind kein Luxus, denn die CMOS-ICs sollten erst zum Schluß eingesetzt werden, um deren Zerstörung durch statische Elektrizität zu vermeiden.
Ist die Platine fertig bestückt, wird der Phototransistor über ein längeres abgeschirmtes Kabel angeschlossen. Die Transistorbasis ist nicht belegt. Dieses Anschlußbeinchen wird gekürzt.
Der Phototransistor hat eine Linse. Der Öffnungswinkel beträgt nur 20°. Daher wird schräg einfallendes Licht nicht gesehen. Wegen der Bildschärfe muß man diesen Blickwinkel noch weiter einengen. Normalerweise wird mit einem Pixelabstand von ca. 1mm gescannt.
Dazu baut man den Phototransistor z.B. in eine Kugelschreiberhülse oder ähnliches ein. Das Öffnungsloch an der Spitze sollte zwischen 0.5 und 1 mm groß sein. Der Phototransistor hat dabei einen Mindestabstand von ca. 10mm zur Öffnung. Grundsätzlich gilt, daß kleine Vorlagen auch kleine Öffnungen benötigen. Ist die Öffnung aber zu klein, verschwinden u.U wichtige Bilddetails in den Abtastzwischenräumen. U.U. ist dann der Phototransistor BPW 13 (=BPW 14 ohne Linse) besser geeignet.
Alternativ kann der Phototransistor auch in eine ausgediente Farbbandkassette eingepaßt werden. Man bohrt an geeigneter Stelle ein kleines Loch mit oben erwähntem Durchmesser in die Kassette.
Die Abtastöffnung muß einerseits möglichst nahe an der Vorlage angebracht sein, andererseits darf sie nicht das Abtastlicht unterbrechen.Kunststoffe sind meistens etwas lichtdurchlässig. Deshalb sollte man das Ganze mit schwarzem Isolierband ©umwickeln bzw. schwarz lackieren. Wenn man das Öffnungsloch verschließt, darf die Schaltung nicht mehr auf das Außenlicht reagieren.
Wenn der Phototransistor nicht genügend gegen Fremdlicht abgeschirmt wird, erhält man unscharfe Bilder. Es empfiehlt sich also, die oben erwähnten Tips beachten.



Das Platinenlayout im Maßstab 1:1

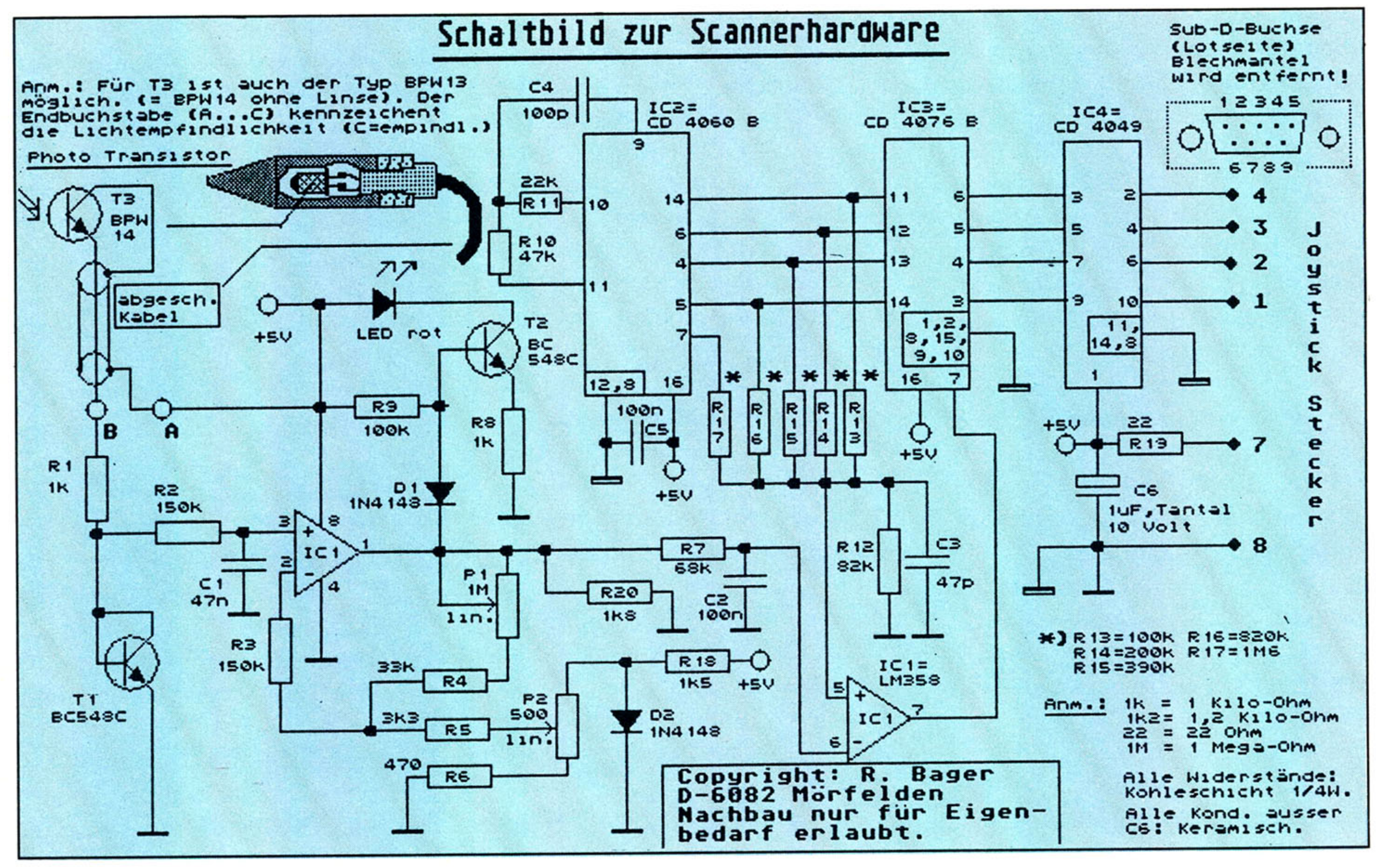


Bild 1: Das Schaltbild

Die Software

Die Software wurde in GFA-BASIC erstellt. Das Scanner-Programm ist Public Domain - also für jeden zugänglich.

Wie bereits erwähnt, wird die Scanner-Hardware am Joystick-Port betrieben. Mein Scannerprogramm fragt diesen Port mit einem ganz ordinären "Peek"-Befehl ab. Das mag zwar die GEM-Spezialisten etwas schockieren, aber das gibt uns die Möglichkeit, auch andere Hardware durch Ändern dieser Peek-Adresse an anderen Ports einzusetzen.

Zum Scannen muß man einen Druckertreiber laden und diesen der Bildvorlage entsprechend modifizieren. Konkret ist wohl stets die Schrittzahl horizontal und vertikal an die jeweiligen Vorlagen anzupassen. Die Schrittweite (Pixelabstände) sollte ebenfalls auf die Vorlagengröße eingestellt werden.

Wenn alle übrigen Vorbereitungen getroffen sind, kann die Abtastung gestartet werden. Auf dem Bildschirm sieht man Zeile um Zeile ein unscharfes "Urbild" entstehen. Der Abtastvorgang kann, falls erforderlich, jederzeit mit der ESC-Taste gestoppt werden.

Wenn der Abtastvorgang beendet ist, wird das Urbild abgespeichert. Hat man das getan, kommt die Aufbereitung. Man verändert jetzt Helligkeit, Kontrast, Bildschärfe und zoomt bzw. entzerrt das Bild.

Danach speichert man das aufbereitete Bild als Neochrome- (erst ab Version 2.0) oder Degas-File ab.

Die Malprogramme Degas (und Neochrome) helfen, das Bild weiter zu verbessern. Auch Fotomontagen sind möglich.

Die Bedienung des Scannerprogramms erklärt sich praktisch von selbst. Einige Punkte muß ich jedoch noch erwähnen:

1. Alle Funktionen und Menüpunkte sind mausgesteuert. Zusätzlich kann die rechte Maustaste einige Menüpunkte direkt auf dem Bildschirm auswählen (z.B. aus dem Druckertreiber).

2. Eingaben müssen mit der <Return>-Taste beendet werden.

3. Zusammen mit dem Druckertreiber speichert bzw. lädt man auch alle Bildkorrekturdaten.

4. Der eigentliche Druckertreiber (im Menü ganz oben) schickt folgende vier unterschiedlichen Zeichenketten an den Drucker:

4.1 Zuerst wird der Drucker initialisiert. Dies geschieht einmalig vor der Abtastung und stellt Geschwindigkeit, Vorschübe oder ähnliches ein. Diese Zeichenkette wird unter dem Menüpunkt "Initialisieren" eingegeben.

4.2 Während der Abtastung wird für jeden horizontalen Vorschub die 2. Zeichenkette ("Hor. Vorschub") ausgegeben.

geht weiter...

4.3 Für jede neue Zeile muß ein Wagenrücklauf und ein Zeilenvorschub erfolgen. Das erledigt die dritte Zeichenkette. ("Vert. Vorschub")

4.4 Die vierte Zeichenkette soll den Drucker wieder in den normalen Betriebszustand zurückstellen. Sie wird unter "Abschluß" eingegeben.

Um "Bildstörungen" bzw. Unschärfe zu verhindern, sind zwei einstellbare Verzögerungszeiten (Abtastung horizontal und vertikal) vorhanden.
Der Treiber wird wie folgt ediert:

<'> Dient nur zur optischen Trennung einzelner Zeichen. Die Zeichenketten werden da durch etwas übersichtlicher. Dieses Zeichen wird später übergangen.

<&> weist das Programm an, die nächsten zwei Zeichen als einen hexadezimalen ASCII-Code zu betrachten.

Beispiel:
Eingabestring "T'ex&74'&0D" wird zum Ausgabestring "Text" + <Wagenrücklauf>.

Scanner - Praxis

1. Ausleuchten der Vorlage

Der verwendete Phototransistor BPY 14 reagiert auf sichtbares Licht. Eine gewöhnliche Tischlampe sorgt für ausreichende Belichtung der Vorlage. Der Phototransistor selbst hat nämlich keine eingebaute Lichtquelle.
Beim Ausleuchten ist unbedingt folgendes zu beachten:

- Die Vorlage muß gleichmäßig ausgeleuchtet werden. Die Ausleuchtung wird durch Abtasten eines weißen Blatt Papiers kontrolliert. Hierbei stellt man in Blattmitte mit dem Abtasthelligkeitsregler an der kleinen Elektronik etwa 8 Helligkeitspunkte ein und fährt anschließend an die Papierränder. Die Helligkeitswerte dürfen sich nur unwesentlich verändern.

- Fremdlicht muß man abschirmen. Ein nahes Fenster beleuchtet mit. Wenn nun plötzlich Wolken aufziehen, kommt es zu einem horizontalen Schattenstreifen.

- Reflexionswinkel unbedingt vermeiden. Die oftmals glänzenden Vorlagen reflektieren wie ein Spiegel. Schlimmstenfalls sieht unser Scanner dann nur noch eine Glühbirne bzw. Neonröhre.

- Beim Flachbettplotter hat es sich bewährt, die Photozelle um 45° zu neigen. Dadurch wirft sich die Photozelle selbst keinen Schatten. Die Belichtungslampe hat einen Abstand von ca. 80 cm und steht senkrecht über der Vorlage.

- Beim Drucker gibt es weniger Reflexionsprobleme, denn auf der gewölbten Vorlage konzentriert sich die Reflexion auf einer Linie. Die Lichtquelle (senkrecht über der Walze) darf wesentlich näher über der eingespannten Vorlage sein. 30 cm Abstand genügen zur gleichmäßigen Ausleuchtung.



2. Einstellen von Abtasthelligkeit und Kontrast

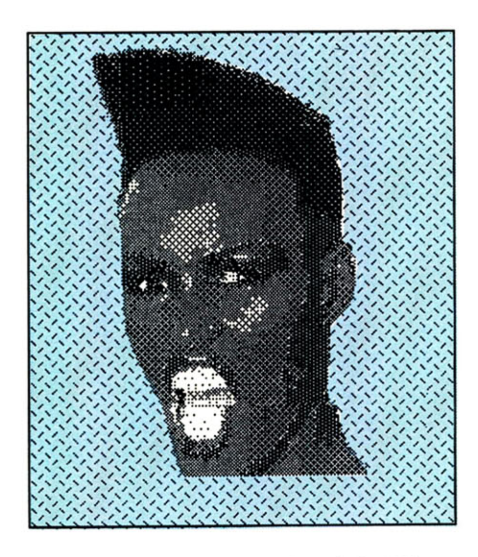
Wie bereits in der Schaltungsbeschreibung erwähnt, bietet schon die Elektronik eine Möglichkeit die Abtasthelligkeit und den Abtastkontrast zu verstellen. Diese Einstellregler müssen immer der jeweiligen Vorlage angepaßt werden.
Helligkeit und Kontrast verhalten sich genau so, wie man es von einem Fernsehgerät her kennt:
Zu wenig Kontrast läßt das Bild grau in grau erscheinen (hellgrau war weiß, dunkelgrau war schwarz). Bei zu viel Kontrast wirkt das Bild "hart". Es fehlen die Mitteltöne (dunkelgrau wird zu schwarz und hellgrau wird zu weiß).
Der Helligkeitsregler hebt oder senkt die Helligkeit aller Bildinhalte gleichmäßig.
Zum Angleich der Regler an die Vorlage geht man zunächst von der Mittelstellung aus. Danach muß zuerst die Helligkeit (P2) und dann der Kontrast (P1) eingestellt werden. Man wiederholt die Einstellung von Helligkeit und Kontrast mehrmals, bis für die hellste Bildpartie (weiß) die Zahl 14 und für die dunkelste die Zahl 1 angezeigt werden.
Die Helligkeitszahl wird rechts unten im Druckermenü ständig angezeigt. Hierbei bedeutet die Zahl 0 das dunkelste Schwarz und die Zahl 15 das hellste Weiß. Es ist also auch möglich, den vollen Zahlenbereich auszuschöpfen indem man die Regler auf 0 bzw. 15 einstellt. Dabei ist es jedoch nicht immer einfach, die Einstellung zu kontrollieren, da ein Weiterdrehen der Regler keine Änderung (kleiner als Null oder größer als 15) bewirkt. Beim Abgleichen von Helligkeit und Kontrast kommt es darauf an, daß die Photozelle auch tatsächlich auf einen sehr hellen bzw. dunklen Bildpunkt

gerichtet ist. Hat man diese beiden Stellen im Bild ausfindig gemacht, wird während des Abgleichs immer zwischen diesen beiden Stellen hin und her gefahren, und dabei wird mit den Reglern auf 14 (wenn hell) und 1 (wenn dunkel) abgestimmt.
Erfahrungsgemäß ist der Kontrastabgleich unkritischer als der Helligkeitsabgleich. Eine Abweichung von +-1 Helligkeitszahl ist nicht so schlimm, denn das fertig abgetastete Bild läßt sich in Helligkeit und Kontrast nachträglich verändern. Der Menüpunkt hierfür heißt "Umfang".

3. Einstellen der Bildformate (Druckertreiber)

Vor der Abtastung muß man sich über die gewünschte Bildgröße auf dem Bildschirm im klaren sein.
Zwar bietet meine Software auch eine Möglichkeit, das Bildformat nachträglich zu verändern bzw. zu verzerren. Aber die besten Ergebnisse bekommt man, wenn man das Bild etwa in der Originalgröße beläßt.
Also zückt man besser vorher den Taschenrechner:
Sicher weiß jeder, daß die monochrome Auflösung 640 x 400 Pixel beträgt. Grundsätzlich sollte man für etwa vier Monitorpixel einen Abtastschritt vorsehen. Soll z.B. das Monitorbild vollständig "gefüllt" werden, wären das etwa 320 Abtastpunkte horizontal und 200 Punkte vertikal. (Zwei horizontale mal zwei vertikale Bildpixel kommen auf ein Urpixel). Für den Preis eines unscharfen Bildes verkürzt sich die Abtastzeit, wenn man über dieses Verhältnis hinausgeht. Ein Erhöhen der Abtastpixelzahl bringt hingegen keine nennenswerte Verbesserung der Bildqualität mehr.
Die USM-Routinen in meinem Programm sind bei einem Urpixel/Bildpixel-Verhältnis von 1:4 optimal. USM bringt augenscheinlich die volle Schärfe wieder zurück.
An dem folgenden Beispiel möchte ich erklären, wie der Druckertreiber eingestellt werden muß:

- Ein Bild von der Größe 100 * 100 mm soll auf 150 * 150 Bildschirmpixel gescannt werden.



- Dazu müssen 75 * 75 Abtast-Pixel (=Urpixel) eingescannt werden.
- Die Schrittweite horizontal beträgt 100 mm / 75 = 1.33 mm
- Die Schrittweite vertikal beträgt ebenfalls 1.33 mm
- Der Druckertreiber wird entsprechend der obigen Vorgabe eingestellt. Horizontal- und Vertikalvorschub lassen sich auf 1.4 mm einstellen. Das ist nahe genug am Wunschwert 1.33 mm. Das abgetastete Bild fällt also etwas kleiner als gewünscht aus. Mit der Option "Zoom" kann das Wunschformat später genau erreicht werden.
- Da das Urpixel etwas größer geraten ist, benötigt man auch weniger Abtast-Schritte: 100 mm / 1.4 mm = ca. 71 Schritte
- Die Schrittzahlen sind unter Schrittzahl horizontal bzw. vertikal einzugeben.
- Die Abtastöffnung der Photozelle wählt man etwas kleiner als die Schrittweite (Hier also ca. 1mm Durchmesser).
- Die Steuercodes für den Druckertreiber werden für die Abtastschrittweite 1,4 mm ediert. Diese Zeichenkette paßt den Drucker- (bzw. Plotter-) Typ an.

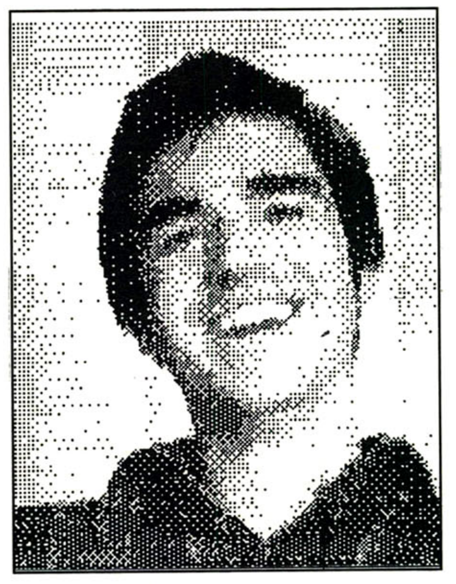
4. Aufbereiten und Verschönern des Urbildes

Die Abtastbilder (Urbilder) können auf Diskette gesichert werden. Vielleicht befinden sich schon einige auf Ihrer PD-Diskette. Diese lassen sich betrachten, indem man sie mit "Lade Urbild" lädt und mit "Zeige Urbild" darstellt.
Das Urbild ist noch unscharf und Gradation, Helligkeit und Kontrast entsprechen noch dem Urzustand. Jedes Urbild kann aber mit den folgenden Optionen verbessert werden:

- USM breit: Normalerweise sind Eigaben von ca. 5 praxisgerecht. Gibt man 0 ein, bleibt die Schärfe auf Urbildniveau. "USM breit" arbeitet auf Urpixel-Ebene und verstärkt dünne Linien und kleine Punkte. Es funktioniert nur, wenn die Abtastöffnung der Photozelle stimmt (Sie muß etwas kleiner als die Abtastschrittweite sein).
- USM schmal: Auf Werte um die Zahl 5 stellen. "USM schmal" arbeitet auf Bildschirmpixel-Ebene (also 1/4 Urpixel) und verhindert u.a. die "Treppchen-Effekte" an diagonal verlaufenden Kanten. Auch hier muß die Breite der Abtastöffnung stimmen.
- Umfang: Funktioniert analog zu Helligkeit und Kontrast mit dem Vorteil, daß sich die beiden Einstellungen nicht beeinflussen. Licht wirkt nicht auf dunkle Bildinhalte und umgekehrt. Wenn Abtastkontrast und -helligkeit gestimmt haben, gibt man hier die Zahl 0 ein, ansonsten zwischen +-5.

- Gradation: Ähnlich wie Umfang. Diese Einstellungen wirken aber nur auf ganz bestimmte Helligkeitsbereiche. Normalerweise stellt man 0 ein. Falls erforderlich, gibt man Werte bis +-5 ein. Positive Zahlen erhöhen stets die Helligkeit und umgekehrt. Die Gradation verändert hellere, mittelhelle und dunkle Grautöne unabhängig voneinander.

Ich wünsche noch viel Spaß beim Basteln und hoffe, daß mein kleines Programm Verwendung findet.



Der Autor selbst

Ein Listing des Source-Codes kann man bei mir anfordern (ca. DM 20,-).

Ralf Bager
Friedensstr. 10 b
6082 Mörfelden-Walldorf

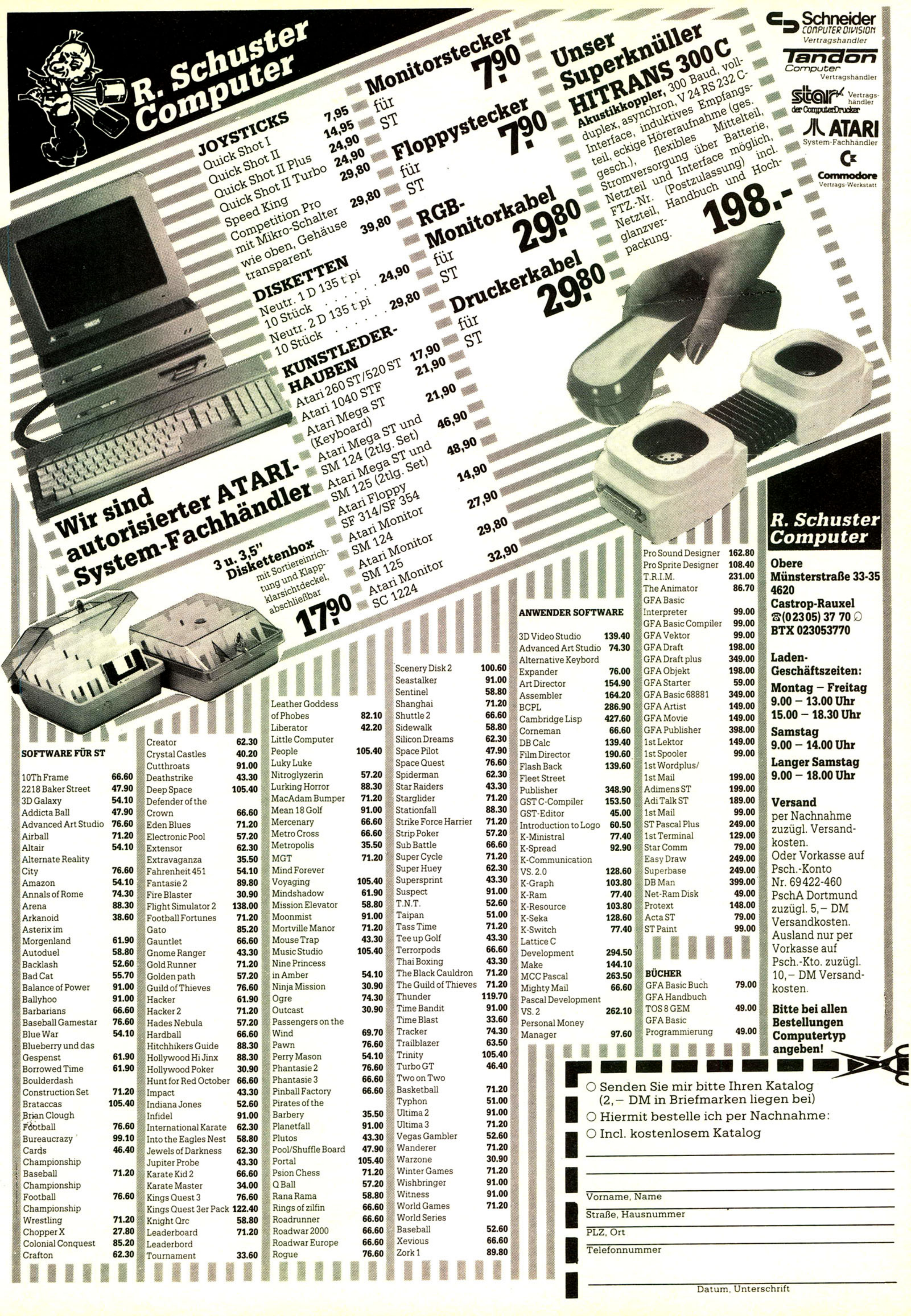
PS: Es gibt bekanntlich verschiedene TOS-Versionen. Deshalb ist es z.B. beim neuen Blitter-TOS oder beim uralten Disketten-TOS erforderlich, die "Joystick Peek-Adresse" zu ändern. Sonst tappt Dein Scannner im Dunkeln. Man kann diese Funktion mit einem normalen Joystick überprüfen.

```
  1:  Dim T$(65),Dr$(65)
  2:  Dim W(65)
  3:  Dim Wert%(256)
  4:  @Men_init
  5:  @Full_init
  6:  @Anz_init
  7:  @Do_var
  8:  Urbild$=Space$(32767)
  9:  Bild$=Space$(32767)
 10:  Eflag=0
 11:  W(23)=3593
 12:  T$(23)=Str$(W(23))
 13:  '
 14:  '
 15:  Start:
 16:  @Tx
 17:  Menu Rollo$()
 18:  On Menu  Gosub Menue
 19:  '
 20:  '
 21:  Do
 22:    On Menu
 23:    If Menu(12)=2
 24:      Repeat
 25:        On Menu
 26:      Until Menu(12)=0
 27:      @Mausklick(Menu(10),Menu(11))
 28:    Endif
 29:    Joy2=Peek(W(23))
 30:    If Joy2<>Joy1
 31:      Joy1=Joy2
 32:      Print At(58,24);Joy1;" "
 33:    Endif
 34:  Loop
 35:  '
 36:  '
 37:  Procedure Mausklick(X,Y)
 38:    Menu Off
 39:    Local I
 40:    X=Int(X/8)
 41:    Y=Int(Y/16)+1
 42:    For I=27 To 65
 43:      If Anz%(0,I)+38>X And Anz%(0,I)<X And Anz%(1,I)=Y
 44:        Mreq=I
 45:        @Menue
 46:        Mreq=0
 47:      Endif
 48:    Next I
 49:  Return
 50:  '
 51:  '
 52:  Procedure Menue
 53:    If Mreq=0
 54:      Me=Menu(0)
 55:    Else
 56:      Me=Mreq
 57:    Endif
 58:    On Me Gosub About
 59:      On Me-10 Gosub N,Drlad,Urlad,N,Drspei,Urspei,Binspei,
            Neospei,Degspei
 60:      On Me-20 Gosub Kopie,N,Zin,N,N,N,Txin,Txin,Txin,Txin
 61:      On Me-30 Gosub Zin,Zin,N,Zin,Zin,N,N,Zoom,N,Zin
 62:      On Me-40 Gosub Zin,N,Zin,Zin,Zin,N,Zin,Zin
 63:      On Me-50 Gosub Bildcalc,Scan,N,N,N,Par_bild,Scan_bild,
            N,Ende
 64:    Menu Off
 65:  Return
 66:  '
 67:  '
 68:  Procedure About
 69:    Local A
 70:    Alert 0,"* * S C A N N E R * *|C.  1987|Ralf Bager,
          Mörfelden",1,"OK",A
 71:  Return
 72:  '
 73:  '
 74:  Procedure Drlad
 75:    Local Du$
 76:    Fileselect "\*.DR1","",Nam$
 77:    If Nam$<>""
 78:      If Exist(Nam$)
 79:        Open "I",#1,Nam$
 80:        For I=10 To 65
 81:          If I<>34 And I<>35
 82:            Line Input #1,T$(I)
 83:          Else
 84:            Line Input #1,Du$
 85:          Endif
 86:        Next I
 87:        Close #1
 88:      Else
 89:        Alert 3,"Datei nicht vorhanden !| | ",1,"OK",Du
 90:      Endif
 91:    Endif
 92:    @Tx
 93:  Return
 94:  '
 95:  '
 96:  Procedure Urlad
 97:    Local Len
 98:    Fileselect "\*.UR1","",Nam$
 99:    If Nam$<>""
100:      If Exist(Nam$)
101:        Urbild$=Space$(32767)
102:        Bload Nam$,Varptr(Urbild$)
103:        Len=Cvl(Left$(Urbild$,4))
104:        T$(34)=Mid$(Urbild$,5,4)
105:        T$(35)=Mid$(Urbild$,9,4)
106:        Urbild$=Mid$(Urbild$,13,Len)
107:      Else
108:        Alert 3,"Datei nicht vorhanden !| | ",1,"OK",Du
109:      Endif
110:    Endif
111:    @Tx
```

geht weiter...

```
112:  Return
113:  '
114:  '
115:  Procedure Urspei
116:    Local Len
117:    Fileselect "\*.UR1","",Nam$
118:    If Nam$<>""
119:      Len=Len(Bild$)
120:      Urbild$=Mkl$(Len(Urbild$))+Right$("      "+T$(34),4)+
            Right$("      "+T$(35),4)+Urbild$
121:      Bsave Nam$,Varptr(Urbild$),Len+12
122:      Urbild$=Mid$(Urbild$,13,Len)
123:    Endif
124:  Return
125:  '
126:  '
127:  Procedure Drspei
128:    Fileselect "\*.DR1","",Nam$
129:    If Nam$<>""
130:      Open "O",#1,Nam$
131:      For I=10 To 65
132:        Print #1,T$(I)
133:      Next I
134:      Close #1
135:    Endif
136:  Return
137:  '
138:  '
139:  Procedure Kopie
140:    @All
141:    If Erg=2
142:      Hardcopy
143:    Endif
144:  Return
145:  '
146:  '
147:  Procedure Scan
148:    Local A$
149:    Cls
150:    @All
151:    If Erg=2
152:      @Dis1
153:      @Do_var
154:      @Full_lut
155:      Cls
156:      @Do_scan
157:    Endif
158:    Repeat
159:      A$=Inkey$
160:    Until A$=Chr$(27)
161:    Cls
162:    Menu Rollo$()
163:    @Tx
164:  Return
165:  '
166:  '
167:  Procedure Ende
168:    Menu Kill
169:    End
170:  Return
171:  '
172:  '
173:  Procedure Zin
174:    Local M
175:    M=Me
176:    @Textedit(Anz%(0,M)+18,Anz%(1,M),4,T$(M),-1)
177:    T$(M)=Rueck$
178:  Return
179:  '
180:  '
181:  Procedure Txin
182:    Local M
183:    M=Me
184:    @Textedit(Anz%(0,M)+18,Anz%(1,M),60,T$(M),0)
185:    T$(M)=Rueck$
186:  Return
187:  '
188:  '
189:  Procedure Bildcalc
190:    Local X,Y,P,B,A$
191:    Cls
192:    @Dis1
193:    @Do_var
194:    @Full_lut
195:    Cls
196:    @Rahmung(0)
197:    For Y=0 To (W(35)-1)
198:      For X=0 To (W(34)-1)
199:        P=(Y*W(34)+X)/2
200:        B=Asc(Mid$(Urbild$,Int(P),1))
201:        If P-Int(P)>0.1
202:          B=(B And 240)/16
203:        Else
204:          B=B And 15
205:        Endif
206:        @Putpix(X*2,Y*2,2,2,B*16)
207:      Next X
208:    Next Y
209:    @Rahmung(1)
210:    Repeat
211:      A$=Inkey$
212:    Until A$=Chr$(27)
213:    Cls
214:    Menu Rollo$()
215:    @Tx
216:  Return
217:  '
218:  '
219:  Procedure Par_bild
220:    Cls
221:    Menu Rollo$()
222:    @Tx
223:  Return
224:  '
225:  '
226:  ' ------------------------------
227:  ' ab hier kommen untergeordnete Proceduren
228:  ' ------------------------------
229:  '
230:  '
231:  Procedure Tx
232:    Local M
233:    For M=0 To 65
234:      If Anz%(0,M)>0
235:        @Str_clean(Rollo$(M))
236:        Print At(Anz%(0,M),Anz%(1,M));Rueck$;":"
237:        Print At(Anz%(0,M)+18,Anz%(1,M));T$(M);
238:      Endif
239:    Next M
240:  Return
241:  '
242:  '
243:  Procedure Str_clean(A$)
244:    Local B$,I
245:    For I=Len(A$) To 1 Step -1
246:      B$=Mid$(A$,I,1)
247:      Exit If B$<>"-" And B$<>" " And B$<>":"
248:    Next I
249:    A$=Left$(A$,I)
250:    If Left$(A$,1)<>"-"
251:      Rueck$=Left$(A$+"...............",17)
252:    Else
253:      Rueck$=A$
254:    Endif
255:  Return
256:  '
257:  '
258:  Procedure Do_var
259:    Local I,K,A$,B$,C$
260:    Eflag=1
261:    If W(34)*W(35)/2>32767
262:      Error 10
263:    Endif
264:    For I=0 To 65
265:      A$=Rollo$(I)
266:      B$=T$(I)
267:      If Len(A$)>0 And Anz%(0,I)<>0 And Left$(A$,1)<>"-"
268:        If I<27 Or I>30
269:          W(I)=Val(B$)
270:        Else
271:          Dr$(I)=""
272:          For K=1 To Len(B$)
273:            C$=Mid$(B$,K,1)
274:            If C$<>"'"
275:              If C$<>"&"
276:                Dr$(I)=Dr$(I)+C$
277:              Else
278:                Dr$(I)=Dr$(I)+Chr$(Val(Mid$(B$,K,3)))
279:                K=K+2
280:              Endif
281:            Endif
282:          Next K
```

geht weiter...



R. Schuster Computer
Schneider COMPUTER DIVISION Vertragshändler
Tandon Computer Vertragshändler
star Vertrags-händler der ComputerDrucker
ATARI System-Fachhändler
Commodore Vertrags-Werkstatt
JOYSTICKS
Quick Shot I 7,95
Quick Shot II 14,95
Quick Shot II Plus 24,90
Quick Shot II Turbo 24,90
Speed King 29,80
Competition Pro mit Mikro-Schalter 29,80
wie oben, Gehäuse transparent 39,80
DISKETTEN
Neutr. 1 D 135 tpi 10 Stück 24,90
Neutr. 2 D 135 tpi 10 Stück 29,80
KUNSTLEDER-HAUBEN
Atari 260 ST/520 ST 17,90
Atari 1040 STF 21,90
Atari Mega ST (Keyboard) 21,90
Atari Mega ST und SM 124 (2tlg. Set) 46,90
Atari Mega ST und SM 125 (2tlg. Set) 48,90
Atari Floppy SF 314/SF 354 14,90
Atari Monitor SM 124 27,90
Atari Monitor SM 125 29,80
Atari Monitor SC 1224 32,90
Monitorstecker für ST 7.90
Floppystecker für ST 7.90
RGB-Monitorkabel für ST 29.80
Druckerkabel für ST 29.80
Unser Superknüller HITRANS 300 C
Akustikkoppler, 300 Baud, vollduplex, asynchron, V 24 RS 232 C-Interface, induktives Empfangsteil, eckige Höreraufnahme (ges. gesch.), flexibles Mittelteil, Stromversorgung über Batterie, Netzteil und Interface möglich, FTZ.-Nr. (Postzulassung) incl. Netzteil, Handbuch und Hochglanzverpackung. 198.-
Wir sind autorisierter ATARI-System-Fachhändler
3 u. 3,5" Diskettenbox mit Sortiereinrichtung und Klappklarsichtdeckel, abschließbar 17.90
SOFTWARE FÜR ST
10Th Frame 66.60
2218 Baker Street 47.90
3D Galaxy 54.10
Addicta Ball 47.90
Advanced Art Studio 76.60
Airball 71.20
Altair 54.10
Alternate Reality City 76.60
Amazon 54.10
Annals of Rome 74.30
Arena 88.30
Arkanoid 38.60
Asterix im Morgenland 61.90
Autoduel 58.80
Backlash 52.60
Bad Cat 55.70
Balance of Power 91.00
Ballyhoo 91.00
Barbarians 66.60
Baseball Gamestar 76.60
Blue War 54.10
Blueberry und das Gespenst 61.90
Borrowed Time 61.90
Boulderdash Construction Set 71.20
Brataccas 105.40
Brian Clough Football 76.60
Bureaucrazy 99.10
Cards 46.40
Championship Baseball 71.20
Championship Football 76.60
Championship Wrestling 71.20
Chopper X 27.80
Colonial Conquest 85.20
Crafton 62.30
Creator 62.30
Crystal Castles 40.20
Cutthroats 91.00
Deathstrike 43.30
Deep Space 105.40
Defender of the Crown 66.60
Eden Blues 71.20
Electronic Pool 57.20
Extensor 62.30
Extravaganza 35.50
Fahrenheit 451 54.10
Fantasie 2 89.80
Fire Blaster 30.90
Flight Simulator 2 138.00
Football Fortunes 71.20
Gato 85.20
Gauntlet 66.60
Gnome Ranger 43.30
Gold Runner 71.20
Golden path 57.20
Guild of Thieves 76.60
Hacker 61.90
Hacker 2 71.20
Hades Nebula 57.20
Hardball 66.60
Hitchhikers Guide 88.30
Hollywood Hi Jinx 88.30
Hollywood Poker 30.90
Hunt for Red October 66.60
Impact 43.30
Indiana Jones 52.60
Infidel 91.00
International Karate 62.30
Into the Eagles Nest 58.80
Jewels of Darkness 62.30
Jupiter Probe 43.30
Karate Kid 2 66.60
Karate Master 34.00
Kings Quest 3 76.60
Kings Quest 3er Pack 122.40
Knight Orc 58.80
Leaderbord 71.20
Leaderbord Tournament 33.60
Leather Goddess of Phobes 82.10
Liberator 42.20
Little Computer People 105.40
Luky Luke
Nitroglyzerin 57.20
Lurking Horror 88.30
MacAdam Bumper 71.20
Mean 18 Golf 91.00
Mercenary 66.60
Metro Cross 66.60
Metropolis 35.50
MGT 71.20
Mind Forever Voyaging 105.40
Mindshadow 61.90
Mission Elevator 58.80
Moonmist 91.00
Mortville Manor 71.20
Mouse Trap 43.30
Music Studio 105.40
Nine Princess in Amber 54.10
Ninja Mission 30.90
Ogre 74.30
Outcast 30.90
Passengers on the Wind 69.70
Pawn 76.60
Perry Mason 54.10
Phantasie 2 76.60
Phantasie 3 66.60
Pinball Factory 66.60
Pirates of the Barbery 35.50
Planetfall 91.00
Plutos 43.30
Pool/Shuffle Board 47.90
Portal 105.40
Psion Chess 71.20
Q Ball 57.20
Rana Rama 58.80
Rings of zilfin 66.60
Roadrunner 66.60
Roadwar 2000 66.60
Roadwar Europe 66.60
Rogue 76.60
Scenery Disk 2 100.60
Seastalker 91.00
Sentinel 58.80
Shanghai 71.20
Shuttle 2 66.60
Sidewalk 58.80
Silicon Dreams 62.30
Space Pilot 47.90
Space Quest 76.60
Spiderman 62.30
Star Raiders 43.30
Starglider 71.20
Stationfall 88.30
Strike Force Harrier 71.20
Strip Poker 57.20
Sub Battle 66.60
Super Cycle 71.20
Super Huey 62.30
Supersprint 43.30
Suspect 91.00
T.N.T. 52.60
Taipan 51.00
Tass Time 71.20
Tee up Golf 43.30
Terrorpods 66.60
Thai Boxing 43.30
The Black Cauldron 71.20
The Guild of Thieves 71.20
Thunder 119.70
Time Bandit 91.00
Time Blast 33.60
Tracker 74.30
Trailblazer 63.50
Trinity 105.40
Turbo GT 46.40
Two on Two Basketball 71.20
Typhon 51.00
Ultima 2 91.00
Ultima 3 71.20
Vegas Gambler 52.60
Wanderer 71.20
Warzone 30.90
Winter Games 71.20
Wishbringer 91.00
Witness 91.00
World Games 71.20
World Series Baseball 52.60
Xevious 66.60
Zork 1 89.80
ANWENDER SOFTWARE
3D Video Studio 139.40
Advanced Art Studio 74.30
Alternative Keybord Expander 76.00
Art Director 154.90
Assembler 164.20
BCPL 286.90
Cambridge Lisp 427.60
Corneman 66.60
DB Calc 139.40
Film Director 190.60
Flash Back 139.60
Fleet Street Publisher 348.90
GST C-Compiler 153.50
GST-Editor 45.00
Introduction to Logo 60.50
K-Ministral 77.40
K-Spread 92.90
K-Communication VS. 2.0 128.60
K-Graph 103.80
K-Ram 77.40
K-Resource 103.80
K-Seka 128.60
K-Switch 77.40
Lattice C Development 294.50
Make 144.10
MCC Pascal 263.50
Mighty Mail 66.60
Pascal Development VS. 2 262.10
Personal Money Manager 97.60
Pro Sound Designer 162.80
Pro Sprite Designer 108.40
T.R.I.M. 231.00
The Animator 86.70
GFA Basic Interpreter 99.00
GFA Basic Compiler 99.00
GFA Vektor 99.00
GFA Draft 198.00
GFA Draft plus 349.00
GFA Objekt 198.00
GFA Starter 59.00
GFA Basic 68881 349.00
GFA Artist 149.00
GFA Movie 149.00
GFA Publisher 398.00
1st Lektor 149.00
1st Spooler 99.00
1st Wordplus/1st Mail 199.00
Adimens ST 199.00
Adi Talk ST 189.00
1st Mail 99.00
ST Pascal Plus 249.00
1st Terminal 129.00
Star Comm 79.00
Easy Draw 249.00
Superbase 249.00
DB Man 399.00
Net-Ram Disk 49.00
Protext 148.00
Acta ST 79.00
ST Paint 99.00
BÜCHER
GFA Basic Buch 79.00
GFA Handbuch TOS 8 GEM 49.00
GFA Basic Programmierung 49.00
R. Schuster Computer
Obere Münsterstraße 33-35
4620 Castrop-Rauxel
(0 23 05) 37 70
BTX 023053770
Laden-Geschäftszeiten:
Montag – Freitag
9.00 – 13.00 Uhr
15.00 – 18.30 Uhr
Samstag
9.00 – 14.00 Uhr
Langer Samstag
9.00 – 18.00 Uhr
Versand
per Nachnahme zuzügl. Versandkosten.
Oder Vorkasse auf Psch.-Konto Nr. 69422-460 PschA Dortmund zuzügl. 5,– DM Versandkosten.
Ausland nur per Vorkasse auf Psch.-Kto. zuzügl. 10,– DM Versandkosten.
Bitte bei allen Bestellungen Computertyp angeben!
○ Senden Sie mir bitte Ihren Katalog (2,– DM in Briefmarken liegen bei)
○ Hiermit bestelle ich per Nachnahme:
○ Incl. kostenlosem Katalog
Vorname, Name
Straße, Hausnummer
PLZ, Ort
Telefonnummer
Datum, Unterschrift

```
283:       Endif
284:      Endif
285:    Next I
286:    Eflag=0
287:  Return
288:  '
289:  '
290:  Procedure Do_scan
291:    @Rahmung(0)
292:    Urbild$=Space$(Int((W(34)*W(35)+1)/2))
293:    Local A,B,X,Y,Z%,A$
294:    Lprint Dr$(27);
295:    Z%=0
296:    For Y=0 To (W(35)-1)
297:      For X=0 To (W(34)-1)
298:        A=B
299:        @Delay(W(31))
300:        B=Peek(Int(W(23)))
301:        Lprint Dr$(28)
302:        If Z%>=1
303:          Z%=0
304:          Mid$(Urbild$,(Y*W(34)+X)/2)=Chr$((A+B*16)
              And 255)
305:        Else
306:          Inc Z%
307:        Endif
308:        @Putpix(X*2,Y*2,2,2,B*16)
309:        A$=Inkey$
310:        Exit If A$=Chr$(27)
311:      Next X
312:      Lprint Dr$(29)
313:      @Delay(W(32))
314:      Exit If A$=Chr$(27)
315:    Next Y
316:    Lprint Dr$(30)
317:    @Rahmung(1)
318:  Return
319:  '
320:  '
321:  Procedure Putpix(X,Y,Dx,Dy,H)
322:    Deffill 1,Muster$(Wert%(H))
323:    Pbox X,Y,X+Dx-1,Y+Dy-1
324:  Return
325:  '
326:  ' bewertet die Füllmuster nach Helligkeit und erzeugt
      Wert%()
327:  Procedure Full_lut
328:    Local H,A,Byte,Bit,M,A$
329:    For A=0 To 256
330:      Wert%(A)=300
331:    Next A
332:    For M=0 To 32
333:      Exit If Muster$(M)=""
334:      H=0
335:      For Byte=1 To 32
336:        A=Asc(Mid$(Muster$(M),Byte,1))
337:        A$=Right$("00000000"+Bin$(A),8)
338:        For Bit=1 To 8
339:          If Mid$(A$,Bit,1)="0"
340:            H=H+1
341:          Endif
342:        Next Bit
343:      Next Byte
344:      Wert%(H)=M
345:    Next M
346:    Repeat
347:      A=1
348:      For H=1 To 255
349:        If Wert%(H)=300 And Wert%(H+1)<>300
350:          Wert%(H)=Wert%(H+1)
351:          A=0
352:        Endif
353:        If Wert%(H)=300 And Wert%(H-1)<>300
354:          Wert%(H)=Wert%(H-1)
355:          A=0
356:          Inc H
357:        Endif
358:      Next H
359:    Until A=1
360:  Return
361:  '
362:  '
363:  Procedure All
364:    Alert 2,"Ist der Drucker bereit|und die Anpassung
          gemacht ?",2,"Abbruch|Start",Erg
365:  Return
366:  '
367:  '
368:  Procedure Dis1
369:    Print At(30,11);" ******************* ";
370:    Print At(30,12);" *** Bitte warten *** ";
371:    Print At(30,13);" ******************* ";
372:  Return
373:  '
374:  '
375:  ' ____________________________
376:  ' Ab hier kommen Lader für div Feld-Variablen
377:  ' ____________________________
378:  '
379:  ' liest menu in Rollo$()
380:  Procedure Men_init
381:    Dim Rollo$(67)
382:    For I=0 To 65
383:      Read Rollo$(I)
384:      Exit If Rollo$(I)="***"
385:    Next I
386:    Let Rollo$(I)=""
387:    Let Rollo$(I+1)=""
388:  Return
389:  ' menu text
390:  Data Desk,  About  SCANNER
391:  Data ———————
392:  Data acc1,acc2,acc3,acc4,acc5,acc6,""
393:  Data    I/O   ,-lade———, Druckertreiber , Ur-Bild ,
        -speichere——
394:  Data   Druckertreiber, Ur-Bild, Bild (BIN), Bild(NEO),
        Bild(P01)
395:  Data ————, Hardcopy,————,
        Joyst.-Peek,""
396:  Data Abtastung    ,-Druckertreiber—, Initialisierung ,
        Horiz.-Vorschub
397:  Data   Vert.-Vorsch;CR , Abschlu_, Timer Horizontal,
        Timer Vertikal,-Schrittzahl—, Horizontal,
        Vertikal,""
398:  Data   Aufber., Zoom,-USM———, breit, schmahl,
        -Gradation—, Licht, Mittelton, Schatten
399:  Data -Grundwerte-, Licht, Schatten,""
400:  Data Starte, Aufbereitung , Abtastung,""
401:  Data   Sonstiges ,-zeige——, Parameter  , Bild  ,
        ———, Ende ,""
402:  Data ***
403:  '
404:  ' liest Füllmuster in Muster$()
405:  Procedure Full_init
406:    Dim Muster$(32)
407:    Dim Zeil$(15)
408:    Local A$,V,N,Z
409:    For N=0 To 32
410:      Read A$
411:      Exit If A$="***"
412:      For V=0 To 15
413:        Read A$
414:        Exit If A$="*"
415:        Repeat
416:          A$=A$+A$
417:        Until Len(A$)>=16
418:        Zeil$(V)=Left$(A$,16)
419:      Next V
420:      For Z=V To 15
421:        Zeil$(Z)=Zeil$(Z-V)
422:      Next Z
423:      Muster$(N)=""
424:      For Z=0 To 15
425:        Muster$(N)=Muster$(N)+Mki$(Int(Val("&X"+
            Zeil$(Z))))
426:      Next Z
427:    Next N
428:  Return
429:  ' Hier beginnen die Füllmuster-Daten
430:  Data **
431:  Data 0
432:  Data *,*
433:  Data 0100
434:  Data 0
435:  Data 0001
436:  Data 0
437:  Data *,*
438:  Data 0100
439:  Data 0001
440:  Data *,*
```

geht weiter...

Zwei oder lieber vier Megabyte RAM gefällig ?

! Voilà !

Passend für alle Atari Computer mit den Typenbezeichnungen 260 ST, 520 ST, 520 ST+, 520 STM, 1040 STF bieten wir Ihnen Arbeitsspeicher satt mit überzeugenden Qualitätsvorteilen :

- Die Erweiterung ist in **drei Ausbaustufen** erhältlich
- Der Einsatz von Stecksockeln ermöglicht jederzeit den **Ausbau auf bis zu 4 MB**
- Sie müssen **keine** Leiterbahnen bzw. Widerstände durchtrennen
- Die ausgeklügelte Einstecktechnik macht bei der Montage den **Lötkolben überflüssig**

4 MB Basisplatine mit 0 MB bestückt	DM 238.50
4 MB Basisplatine mit 2 MB bestückt	DM 930.-
4 MB Basisplatine mit 4 MB bestückt	DM 1699.90
passender Steckadapter für 1040 STF	DM 57.-
passender Steckadapter für alle übrigen	DM 64.-
Megabit-RAM's (16 Stück)	DM 798.-

Aufgrund des schnellen Wechsels am Speichermarkt können sich die Preise geändert haben; fragen Sie daher bitte die aktuellen Tagespreise telefonisch an.

!NEU MEGA - CLOCK NEU!

Echtzeituhr für alle "kleinen" ST

- funktionsidentisch zur Uhr im MEGA ST
- mit dem neuen Blitter - TOS ist der Betrieb ohne Software möglich
- Software für das alte TOS wird mitgeliefert
- langzeitbatteriegepuffert
- kann ebenfalls ohne Löten eingebaut werden
- moderne SMD-Technik sorgt für geringsten Platzbedarf

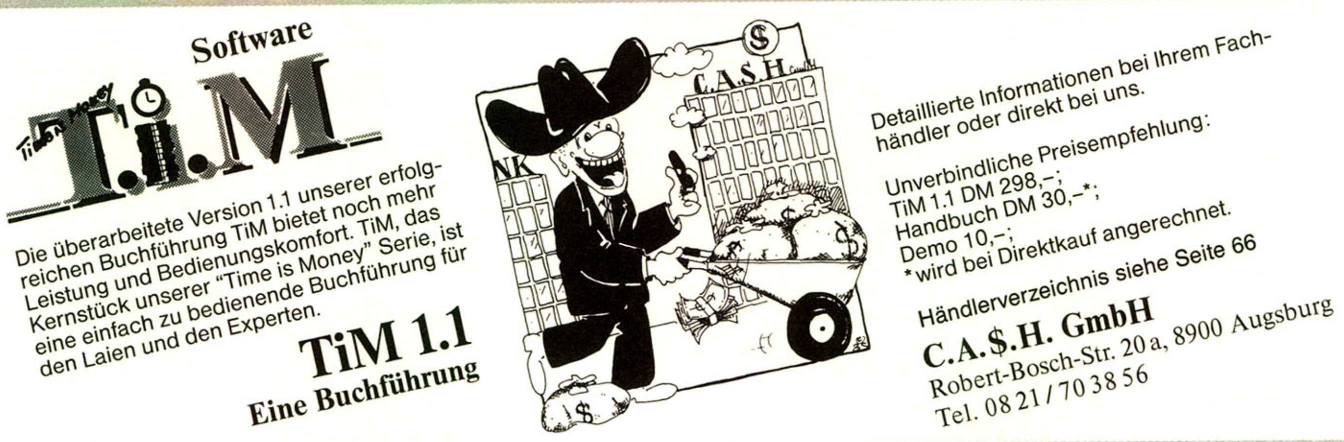
MEGA - Leistung , MINI - Preis DM 96.90

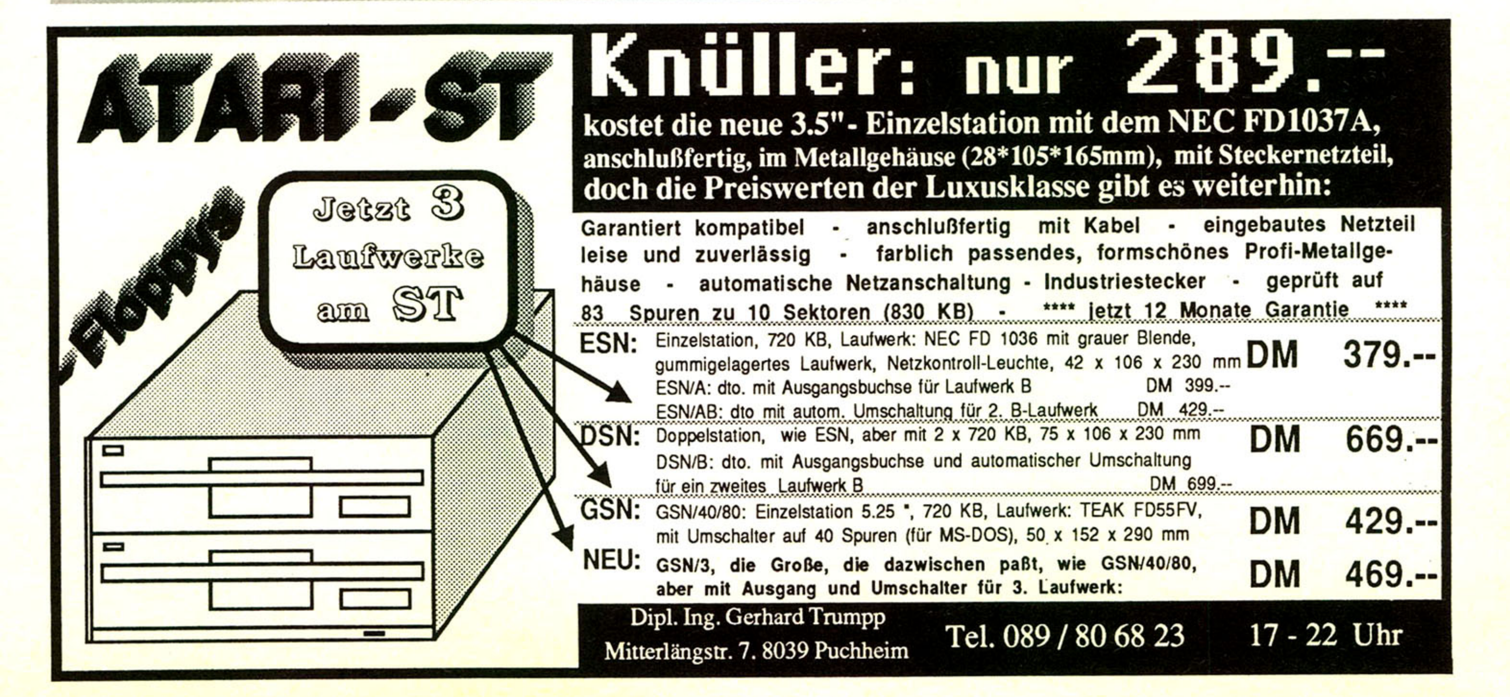
Die Lieferung erfolgt per Nachnahme zuzüglich DM 6,50 Versandkosten. Anfragen und Bestellungen richten Sie bitte an :



Kaiserstraße 5 - 7
5657 Haan 1
Tel. 02129 / 50819







```
441:  Data 0101
442:  Data 0010
443:  Data 1010
444:  Data 0100
445:  Data *,*
446:  Data 01
447:  Data 10
448:  Data *,*
449:  Data 0101
450:  Data 1101
451:  Data 1010
452:  Data 1011
453:  Data *,*
454:  Data 0111
455:  Data 1101
456:  Data *,*
457:  Data 0111
458:  Data 1111
459:  Data 1101
460:  Data 1111
461:  Data *,*
462:  Data 1
463:  Data *,***
464:  '
465:  ' zeigt wo und welche menu-Einträge angezeigt werden
466:  Procedure Anz_init
467:    Dim Anz%(1,65)
468:    Local X,Y,A
469:    For A=0 To 65
470:      Anz%(0,A)=0
471:      Anz%(1,A)=0
472:    Next A
473:    Do
474:      Read A
475:      Exit If A=999
476:      Read X,Y
477:      Anz%(0,A)=X
478:      Anz%(1,A)=Y
479:    Loop
480:  Return
481:  '
482:  Data 23,40,22,  26,30,3,  27,1,4,   28,1,5,   29,1,6,
        30,1,7
483:  Data 31,1,9,    32,40,9,  33,1,12,  34,1,13,  35,1,14,
        39,40,12
484:  Data 40,40,13,  41,40,14, 42,1,17,  43,1,18,  44,1,19,
        45,1,20
485:  Data 46,40,17,  47,40,18, 48,40,19, 999
486:  '
487:  '
488:  '------------------------------
489:  ' Ab hier kommen allgemeine Proceduren
490:  '------------------------------
491:  '
492:  '
493:  Procedure Textedit(X,Y,Lmax,A$,Z)
494:    Local B$,L,P,B,C
495:    L=Len(A$)
496:    P=L+1
497:    Do
498:      Print At(X,Y);A$;" "
499:      Print Chr$(27);"p";
500:      If L>0 And P<=L
501:        Print At(X+P-1,Y);Mid$(A$,P,1);
502:        Print Chr$(27);"q";
503:      Else
504:        Print At(X+P-1,Y);" ";
505:        Print Chr$(27);"q";
506:        Print At(X+P,Y);" ";
507:      Endif
508:      C=Bios(2,2)
509:      B=C And 255
510:      B$=Chr$(B)
511:      C=C/&H10000 And 255
512:      Exit If B=13 Or B=27
513:      If B=127 And L>0 And P<=L
514:        A$=Left$(A$,P-1)+Right$(A$,L-P)
515:        L=Len(A$)
516:      Endif
517:      If B=8 And P>1 And L>0
518:        A$=Left$(A$,P-2)+Right$(A$,L-P+1)
519:        L=Len(A$)
520:        Dec P
521:      Endif
522:      If C=75 And P>1
523:        Dec P
524:      Endif
525:      If C=77 And P<=L
526:        Inc P
527:      Endif
528:      If B=9
529:        P=L+1
530:      Endif
531:      If ((B>31 And B<>127 And Z<>-1) Or (B>44 And B<58
            And B<>47 And Z=-1)) And Lmax>L
532:        If L>=P
533:          A$=Left$(A$,P-1)+B$+Right$(A$,L-P+1)
534:        Else
535:          A$=A$+B$
536:        Endif
537:        L=Len(A$)
538:        Inc P
539:      Endif
540:    Loop
541:    Print At(X,Y);A$;" ";
542:    Rueck$=A$
543:  Return
544:  '
545:  ' leere Procedur
546:  Procedure N
547:  Return
548:  '
549:  '
550:  Procedure Fehler
551:    Local Du
552:    If Eflag=0
553:      Alert 3,"Fehler Nr.:"+Str$(Err)+"| | ",1,"OK",Du
554:      Close
555:    Else
556:      Alert 3,"Fehler im Druckertreiber|oder in
            Korrekturdaten| ",1,"OK",Du
557:    Endif
558:    Eflag=0
559:    Cls
560:    Menu Rollo$()
561:    Resume Start
562:  Return
563:  '
564:  '
565:  Procedure Rahmung(A)
566:    Dpoke Intin,A
567:    Vdisys 104
568:  Return
569:  '
570:  '
571:  Procedure Delay(A)
572:    Pause A
573:  Return
```

ENDE

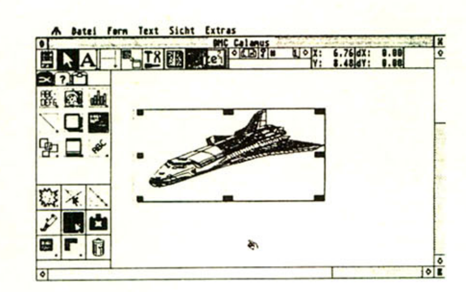
Gegendarstellung

Herr Kleinmeier behauptete, DTP-Programme sind schwierig. Falsch, er kannte "Timeworks DTP" nicht. Herr Kleinmeier behauptete, Tabellenkalkulation ist für einen normalen Anwender unbeherrschbar. Das war so, bevor es "LOGiSTiX" gab. Herr Kleinmeier behauptete, trotz Computer noch Terminkalender, Adreßbuch, Telefonliste und Notizblock zu benötigen. Falsch, er braucht nur "Desk Assist +++". Herr Kleinmmeier glaubte, Firma Kieckbusch würde Endkunden nicht beliefern. Falsch, Kieckbusch liefert an jeden, ohne Ansehen von Ehestand, Religion oder Partei.



■ Timeworks DTP

Das deutsche Timeworks DTP für nur 389,- DM besitzt alle Funktionen, die der Anwender benötigt, um professionelle Vorlagen zu fertigen. Es ist (fast) ohne Handbuch zu bedienen. Im Gegensatz zu sogenannten Page-Programmen besteht beim Timeworks DTP nicht die 1-Seiten-Begrenzung. Der Anwender kann hier große Textmengen (z.B. 50 Seiten) einladen, bearbeiten, formatieren und ausdrucken (z.B. auf P6, Atari-Laser, Postscript-Laser). Der deutsche Distributor, die Firma Kieckbusch, hat eine Aktion laufen: Besitzer von anderen DTP-Programmen (z.B. Becker Page oder Publishing Partner) können dieses für 189,- DM gegen Timeworks umtauschen. Ein, wie wir meinen, ganz interessantes Angebot.



■ Calamus DTP

Das Super-DTP-Programm aus dem Hause DMC findet sich auch im Vertrieb der Firma Kieckbusch. Calamus kostet 998,- DM und läßt nun wirklich keinen Wunsch offen. Es nimmt für sich die Prädikate "intelligent", "komfortabel" und "mächtig" in Anspruch. Vollkommen zu recht, wie wir meinen. Neben dem Programm bietet die Firma Kieckbusch für Calamus auch Schulung an.

■ STEVE

Das integrierte Programm mit den Teilen "Text", "Grafik" und "Datenbank" wird seit ca. 1 Jahr für den Atari ST verkauft. Es ist sehr komplex und wird deshalb komplett mit Handbuch plus Lehrbuch für 398,- DM ausgeliefert. Zur CeBit '88 kommen die Kieckbusch-Leute mit den Versionen 3.0 und 3.0S heraus.

Die Ausgabe 3.0S gestattet das Einlesen von Texten vom Scanner her. Dies funktioniert mit dem A4-Hawk-Scanner wie auch mit dem Handy-Scanner ! Die Erkennung hat mit verschiedenen Schriften eine Erfolgsquote von 97%. Außerdem ist STEVE lernfähig d.h., daß der Anwender seinem Computer mittels STEVE das Lesen jeder Schrift beibringen kann. Endlich muß er die Bild Zeitung nicht mehr selbst lesen ! STEVE tut das für Ihn, selektiert nach Suchkriterium und legt z.B. alle Artikel in der Datenbank ab, die sich mit Grün Kohl, Rot Kohl oder Helmut Kohl befassen. Für nur 1198,- DM ist das möglich.

■ Handy-Scanner

Der kleine Bruder eines richtigen Scanners profiliert sich bei kleineren Anwendungen oder bei speziellen Aufgaben, z.B. beim Einlesen von einem großen Objekt (Whisky-Flasche oder so) oder beim Einlesen von kurzen Passagen aus dicken Büchern, um z.B. in STEVE 3.0S einen Ronald Reagan Zitatenschatz anzulegen.

■ Desk Assist +++

Als unverzichtbar für jeden ernsthaften Anwender hat sich dieses Programm erwiesen. Egal, ob gerade mit Timeworks oder VIP gearbeitet wird, stets sind direkt die Funktionen von "Desk Assist" abrufbar: Rechnen, Adressen suchen, Spooler einrichten, Hardcopy, Kalender/Terminplan, Uhr, Wecker usw. Die brauchbaren Hilfsprogramme "Disk Assist" uns "Term Assist" (Fernübertragung) werden mitgeliefert. Preis 149,- DM.

■ Tabellenkalkulation

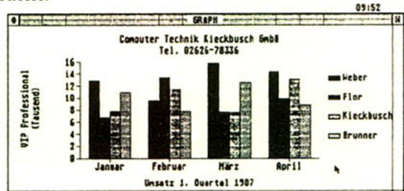
Auf diesem wichtigen Sektor bietet die Firma Kieckbusch für jeden Bedarf das richtige Programm:

□ LOGISTIX

für 399,- DM / für den Einsatz in allen Kalkulationsaufgaben.. Die Möglichkeiten, aus den Zahlenmengen anschauliche Grafiken zu produzieren, sind hier besonders ausgeprägt. Über 50 Drucker- und Plotter Treiber ermöglichen die Ausgabe auf fast jedes Gerät. Außerdem ist eine komfortable Netzplanerstellung mit integriert.

□ VIP Professional

komplett in Deutsch für nur noch 299,- DM ist der Knüller schlechthin. Das Programm ist zu 100% dem Lotus 1-2-3 nachempfunden und läuft zudem unter GEM. Wie die Firma Kieckbusch dieses Preis-Angebot machen kann, ist ein echtes Geheimnis. Niemand weiß, wie lange es durchgehalten werden kann.



■ A4-Hawk-Scanner

Mit diesem Scanner lassen sich ganze Seiten Grafik oder Schrift in ca. 10 Sek. einlesen. Grafiken lassen sich dann nahezu beliebig verändern, Schrift läßt sich mit STEVE 3.0S in ASCII-Text umformen. Beides läßt sich dann z.B. in Timeworks DTP weiter verarbeiten. Der eingebaute A4-Drucker gibt alles schnell und geräuschlos auf Thermopapier aus. Die Qualität der Hardware, verbunden mit einer ausgefeilten Software hat seinen Preis: 2.998,- DM.

☆ Aktionen

Hawk-Scanner inkl. Timeworks: 3048
Hawk-Scanner inkl. Calamus: 3498
Hawk-Scanner inkl.STEVE 3.0S: 3775
Atari Laser inkl. Timeworks: 3498
Atari Laser inkl. Calmus: 3998
Atari Mega ST4 inkl. Steve 3.0: 3998

Gebrauchte DTP-Programme, Becker Page, Fleet Street Editor, Publishing Partner.... je schlappe 50 DM

Beratung, Information, Prospekte von
Computer Technik Kieckbusch GmbH
Baumstammhaus, 5419 Vielbach
☎ 02626-78336 & 8991 FAX -78337

10 MEGAHERTZ PER SOFTWARE

Nach dem Studium von bisher nicht zugänglichen Unterlagen von ATARI können wir Ihnen heute ein Programm vorstellen, das schon bald auf jedem ST eingesetzt werden wird. CLOCK UP erhöht die Taktfrequenz des 68000 im ST auf knapp 10 Megahertz.

Versuche zur Steigerung der Rechenleistung für den ATARI ST liefen bisher über den Einsatz zusätzlicher Hardware. Es werden mehrere Prozessorkarten mit 68020 angeboten, die allerdings keine 100%-ige Kompatibiltät wahren können. Die Erhöhung der Taktfrequenz durch Quarzaustausch scheiterte an den Peripheriebausteinen, die dafür zu langsam sind.

Aufgrund eines Gespräches mit Shiraz Shivji, dem ATARI-Chefentwickler, auf der Comdex Fall in Las Vegas wurden uns inzwischen einige neue Unterlagen über den MFP-Baustein zugänglich gemacht, die eine Überraschung bargen: Die Taktfrequenz der CPU ist durch einfaches Setzen des untersten Bits im FAST-Steuerregister auf 10 Megahertz umschaltbar. Das bisher undokumentierte Register des MFP 68901 liegt bei Adresse $FFFA30 und muß zum Beschleunigen auf $0001 gesetzt werden (siehe Bild 1).

Um den Problemen mit den zu langsamen Peripheriebausteinen aus dem Weg zu gehen, ist der Quarz unterschiedlich mit den CPU und den I/O-Chips verdrahtet. Durch einen Frequenzdiskretor werden MFP, ACIA 1 und 2, DMA, Shifter und Soundchip mit der gewohnten Taktfrequenz von ca. 8MHz betrieben. Eine kleine Lötbrücke auf der Platinenrückseite sorgt dafür, daß der Bus entsprechend synchronisiert wird.

Allerdings ist es damit nicht getan - um die Beschleunigung einsetzen zu können, ist Softwareunterstützung notwendig. Das Problem liegt dabei in den ROM-Chips, bei denen von ATARI aus Kostengründen die langsamere Version gewählt wurde.

Die notwendige Zusatzsoftware muß also vor ROM-Zugriffen dafür sorgen, daß die Taktfrequenz kurzfristig gedrosselt wird. Diese Verlangsamung und die anschließende Beschleunigung sind durch einfaches Umschalten des genannten Steuerbits schnell und problemlos möglich, so daß kein Zeitnachteil bei Betriebssystemaufrufen entsteht.

Das abgedruckte Programm CLOCK UP verbiegt nun alle Trapvektoren für GEMDOS, BIOS, XBIOS und GEM-Aufrufe auf eine kleines Programmstück, das das FST-Bit umschaltet und dann über die alten Vektoren den Call durchführt.

Da das Programm auf dem Papier (Lattice-C und hinzugelinkter MCC-Assemblercode) doch recht lang geraten ist, wollten wir Ihnen mehrere Seiten Listing ersparen und drucken hier einen kleinen Lader für GFA-BASIC ab. Er erzeugt auf Diskette oder Festplatte ein File namens CLOCK UP.ACC, das beim Booten als Accessory geladen wird. Durch die Implementierung als Accessory konnte etwas an Programmcode eingespart werden. Es belegt übrigens keinen Platz in der Accessory-Leiste, bleibt aber automatisch resident im Speicher.

Was bringen nun die 10 Megahertz ? Betriebssystem- und I/O-Zugriffe werden nicht beschleunigt, dafür steigt die CPU-Leistung bei reinen Rechenaufgabe merklich an. Wir haben einige Benchmarks mit den verschiedensten Programmen durchgeführt, deren Ergebnisse Sie in Bild 2 sehen. Die Geschwindigkeitssteigerung um ca. 30 % ist erstaunlich.

Shiraz Shivji antwortete übrigens auf die Frage, warum diese Möglichkeit bis jetzt noch nicht publiziert wurde, damit, daß bei ATARI lange Zeit nicht feststand, welche ROM-Bausteine verwendet werden sollten und gleichzeitig alle Programmierer mit der Implementierung des ROM-TOS beschäftigt waren. Inzwischen habe man feststellen müssen, daß eine Einwicklung der Software im Hause ATARI als zu kompliziert angesehen wird. Jack Tramiel persönlich habe daher verfügt, die Informationen an die Fachpresse zu leiten, um ein Fiasko wie bei ST-Basic zu vermeiden.

Da wir noch sehr wenige Programme auf CLOCK UP-Kompatibilität prüfen konnten, wären wir Ihnen, den Lesern dankbar, wenn Sie über eventuelle Schwierigkeiten mit CLOCK UP berichten würden.

Robert Tolksdorf

Listing siehe nächste Seite...

FAST-Register ($ FFFA30)	×	×	×	×	×	×	×	FST

Bild 1: Das FAST-Register des MFP

Test	normal	CLOCKUP
WordPlus: Suchen + Ersetzen	1:59	1:21
GfA: Fibonaccizahlen errechnen	3:04	2:30
Lattice-C: compilieren	1:20	0:55
TEX: Dokument formatieren	4:23	3:15

Bild 2: Die Benchmarks

```
  1:    ' Loader fuer CLOCKUP -  1.4.88 Robert Tolksdorf
  2:   Open "O",#1,"clockup.acc"
  3:  For I=1 To 1122
  4:    Read A
  5:       Print #1,Chr$(Int(A/256));Chr$(A-Int(A/256)*256);
  6:    C=C+A
  7:  Next I
  8:  If C<>11621566 Then
  9:     Print "Fehler in Datas !"
 10:  Else
 11:    Close #1
 12:  Endif
 13: Data &h601A,&h0000,&h07FA,&h0000,&h0000,&h0000
 14: Data &h074C,&h0000,&h0000,&h0000,&h0000,&h0000
 15: Data &h0000,&h0000,&h4EF9,&h0000,&h0570,&h0000
 16: Data &h2000,&h4E71,&h4E71,&h4E71,&h49F9,&h0000
 17: Data &h0A36,&h2A0C,&h2E45,&h204F,&h42A0,&h42A0
 18: Data &h42A0,&h213C,&h0000,&h1000,&h42A0,&h2A48
 19: Data &h90FC,&h0024,&h43FA,&h0006,&h2109,&h2E48
 20: Data &h9DCE,&h4EF9,&h0000,&h0156,&h4143,&h4358
 21: Data &h0000,&h4D6F,&h6475,&h6C61,&h2D32,&h2F53
 22: Data &h5420,&h2863,&h2920,&h436F,&h7079,&h7269
 23: Data &h6768,&h7420,&h4D6F,&h6475,&h6C61,&h2032
 24: Data &h2053,&h6F66,&h7477,&h6172,&h6520,&h4C74
 25: Data &h642E,&h2031,&h3938,&h352C,&h3139,&h3836
 26: Data &h2C31,&h3938,&h372E,&h2000,&h2863,&h2920
 27: Data &h436F,&h7079,&h7269,&h6768,&h7420,&h5444
 28: Data &h4920,&h536F,&h6674,&h7761,&h7265,&h2049
 29: Data &h6E63,&h2E20,&h3139,&h3835,&h2C31,&h3938
 30: Data &h362C,&h3139,&h3837,&h2E20,&h0000,&h4E56
 31: Data &h0000,&h49F9,&h0000,&h0A6E,&h38AE,&h0010
 32: Data &h396E,&h000E,&h0002,&h396E,&h000C,&h0004
 33: Data &h396E,&h000A,&h0006,&h396E,&h0008,&h0008
 34: Data &h203C,&h0000,&h00C8,&h49F9,&h0000,&h0A38
 35: Data &h2A0C,&h2205,&h4E42,&h33F9,&h0000,&h0A9A
 36: Data &h0000,&h0ABE,&h3D79,&h0000,&h0ABE,&h0012
 37: Data &h4E5E,&h4E75,&h4E56,&h0000,&h33EE,&h000A
 38: Data &h0000,&h0A78,&h33EE,&h0008,&h0000,&h0A7A
 39: Data &h4E5E,&h4E75,&h4E56,&h0000,&h286E,&h0014
 40: Data &h38B9,&h0000,&h0A9C,&h286E,&h0010,&h38B9
 41: Data &h0000,&h0A9E,&h286E,&h000C,&h38B9,&h0000
 42: Data &h0AA0,&h286E,&h0008,&h38B9,&h0000,&h0AA2
 43: Data &h4E5E,&h4E75,&h4EF9,&h0000,&h000A,&h4E56
 44: Data &h0000,&h49F9,&h0000,&h0A50,&h42AC,&h000A
 45: Data &h42AC,&h000E,&h42AC,&h0012,&h42AC,&h0016
 46: Data &h42AC,&h001A,&h49F9,&h0000,&h0A38,&h47F9
 47: Data &h0000,&h0A6E,&h2A0B,&h2885,&h47F9,&h0000
 48: Data &h0A50,&h2A0B,&h2945,&h0004,&h47F9,&h0000
 49: Data &h0A78,&h2A0B,&h2945,&h0008,&h47F9,&h0000
 50: Data &h0A9A,&h2A0B,&h2945,&h000C,&h47F9,&h0000
 51: Data &h0AAA,&h2A0B,&h2945,&h0010,&h47F9,&h0000
 52: Data &h0AB6,&h2A0B,&h2945,&h0014,&h4E5E,&h4EF9
 53: Data &h0000,&h02BE,&h4E56,&h0000,&h558F,&h3F3C
 54: Data &h000A,&h4267,&h3F3C,&h0001,&h4267,&h4267
 55: Data &h4EB9,&h0000,&h00BA,&h4FEF,&h000A,&h3D5F
 56: Data &h0008,&h4E5E,&h4E75,&h4E56,&h0000,&h3F2E
 57: Data &h000A,&h3F2E,&h0008,&h4EB9,&h0000,&h0108
 58: Data &h588F,&h558F,&h3F3C,&h0018,&h3F3C,&h0002
 59: Data &h3F3C,&h0001,&h4267,&h4267,&h4EB9,&h0000
 60: Data &h00BA,&h4FEF,&h000A,&h33DF,&h0000,&h0ABE
 61: Data &h4E5E,&h4E75,&h4E56,&hFFFE,&h558F,&h3F3C
 62: Data &h004D,&h4267,&h3F3C,&h0005,&h4267,&h4267
 63: Data &h4EB9,&h0000,&h00BA,&h4FEF,&h000A,&h3D5F
 64: Data &hFFFE,&h286E,&h0014,&h4854,&h286E,&h0010
 65: Data &h4854,&h286E,&h000C,&h4854,&h286E,&h0008
 66: Data &h4854,&h4EB9,&h0000,&h0120,&h4FEF,&h0010
 67: Data &h3D6E,&hFFFE,&h0018,&h4E5E,&h4E75,&h4E56
 68: Data &h0000,&h7073,&h2239,&h0000,&h0EEC,&h4E42
 69: Data &h4E5E,&h4E75,&h4E56,&h0000,&h33EE,&h0010
 70: Data &h0000,&h0AC0,&h33EE,&h000E,&h0000,&h0AC2
 71: Data &h33EE,&h000C,&h0000,&h0AC6,&h33EE,&h000A
 72: Data &h0000,&h0ACA,&h33EE,&h0008,&h0000,&h0ACC
 73: Data &h4E5E,&h4E75,&h4EF9,&h0000,&h0150,&h4E56
 74: Data &h0000,&h49F9,&h0000,&h0AD8,&h2A0C,&h23C5
 75: Data &h0000,&h0EDC,&h49F9,&h0000,&h0BD8,&h2A0C
 76: Data &h23C5,&h0000,&h0EE0,&h49F9,&h0000,&h0CD8
 77: Data &h2A0C,&h23C5,&h0000,&h0EE4,&h49F9,&h0000
 78: Data &h0DD8,&h2A0C,&h23C5,&h0000,&h0EE8,&h49F9
 79: Data &h0000,&h0AC0,&h2A0C,&h23C5,&h0000,&h0EF0
 80: Data &h23F9,&h0000,&h0EDC,&h0000,&h0EF4,&h23F9
 81: Data &h0000,&h0EE0,&h0000,&h0EF8,&h23F9,&h0000
 82: Data &h0EE4,&h0000,&h0EFC,&h23F9,&h0000,&h0EE8
 83: Data &h0000,&h0F00,&h49F9,&h0000,&h0EF0,&h2A0C
 84: Data &h23C5,&h0000,&h0EEC,&h4E5E,&h4EF9,&h0000
 85: Data &h0576,&h4E56,&h0000,&h33EE,&h0008,&h0000
 86: Data &h0AD8,&h3F3C,&h0020,&h4267,&h3F3C,&h0001
 87: Data &h4267,&h3F2E,&h000A,&h4EB9,&h0000,&h0288
 88: Data &h4FEF,&h000A,&h4EB9,&h0000,&h0276,&h3D79
 89: Data &h0000,&h0CD8,&h000C,&h4E5E,&h4E75,&h4E56
 90: Data &h0000,&h4279,&h0000,&h0BD8,&h33EE,&h0018
 91: Data &h0000,&h0BDA,&h3F3C,&h000C,&h3F3C,&h0001
 92: Data &h4267,&h4267,&h3F2E,&h001A,&h4EB9,&h0000
 93: Data &h0288,&h4FEF,&h000A,&h4EB9,&h0000,&h0276
 94: Data &h286E,&h0014,&h38B9,&h0000,&h0DD8,&h286E
 95: Data &h0010,&h38B9,&h0000,&h0DDA,&h286E,&h000C
 96: Data &h38B9,&h0000,&h0DDC,&h286E,&h0008,&h38B9
 97: Data &h0000,&h0DDE,&h4E5E,&h4E75,&h4E56,&h0000
 98: Data &h3F3C,&h0081,&h3F3C,&h0002,&h3F3C,&h0001
 99: Data &h4267,&h3F2E,&h000E,&h4EB9,&h0000,&h0288
100: Data &h4FEF,&h000A,&h33EE,&h000C,&h0000,&h0AD8
```

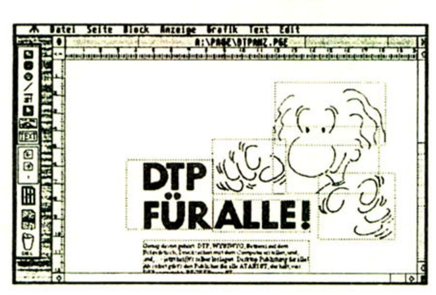
```
101:  Data &h4279,&h0000,&h0F08,&h3A39,&h0000,&h0F08
102:  Data &h286E,&h0008,&hDA45,&h3839,&h0000,&h0F08
103:  Data &hD844,&h47F9,&h0000,&h0BD8,&h37B4,&h5000
104:  Data &h4000,&h5279,&h0000,&h0F08,&h0C79,&h0004
105:  Data &h0000,&h0F08,&h66D0,&h4EB9,&h0000,&h0276
106:  Data &h4E5E,&h4E75,&h4E56,&h0000,&h3F3C,&h0026
107:  Data &h4267,&h4267,&h4267,&h3F2E,&h000C,&h4EB9
108:  Data &h0000,&h0288,&h4FEF,&h000A,&h4EB9,&h0000
109:  Data &h0276,&h4279,&h0000,&h0F0C,&h3A39,&h0000
110:  Data &h0F0C,&hDA45,&h49F9,&h0000,&h0CD8,&h3839
111:  Data &h0000,&h0F0C,&h266E,&h0008,&hD844,&h37B4
112:  Data &h5000,&h4000,&h5279,&h0000,&h0F0C,&h0C79
113:  Data &h0006,&h0000,&h0F0C,&h66D0,&h33FC,&h0006
114:  Data &h0000,&h0F0C,&h3A39,&h0000,&h0F0C,&h5D45
115:  Data &hDA45,&h49F9,&h0000,&h0DD8,&h3839,&h0000
116:  Data &h0F0C,&h266E,&h0008,&hD844,&h37B4,&h5000
117:  Data &h4000,&h5279,&h0000,&h0F0C,&h0C79,&h000A
118:  Data &h0000,&h0F0C,&h66CE,&h4E5E,&h4E75,&h4E56
119:  Data &h0000,&h4279,&h0000,&h0F10,&h33EE,&h000C
120:  Data &h0000,&h0F12,&h3A39,&h0000,&h0F10,&hBA79
121:  Data &h0000,&h0F12,&h6E10,&h3A39,&h0000,&h0F10
122:  Data &h286E,&h0008,&h4A34,&h5000,&h6604,&h4EFA
123:  Data &h002C,&h3A39,&h0000,&h0F10,&h286E,&h0008
124:  Data &h7800,&h1834,&h5000,&h3A39,&h0000,&h0F10
125:  Data &hDA45,&h49F9,&h0000,&h0AD8,&h3984,&h5000
126:  Data &h5279,&h0000,&h0F10,&h60B4,&h3F3C,&h0008
127:  Data &h3F3C,&h0001,&h3F39,&h0000,&h0F10,&h4267
128:  Data &h3F2E,&h0012,&h4EB9,&h0000,&h0288,&h4FEF
129:  Data &h000A,&h33EE,&h0010,&h0000,&h0BD8,&h33EE
130:  Data &h000E,&h0000,&h0BDA,&h4EB9,&h0000,&h0276
131:  Data &h4E5E,&h4E75,&h4EF9,&h0000,&h02B8,&h4E56
132:  Data &h0000,&h558F,&h6100,&hFC4A,&h33DF,&h0000
133:  Data &h0F18,&h4267,&h3F3C,&h0001,&h6100,&hFC60
134:  Data &h588F,&h33FC,&h7FFF,&h0000,&h0F1A,&h3F39
135:  Data &h0000,&h0F1A,&h4267,&h6100,&hFC4A,&h588F
136:  Data &h0C79,&h03E8,&h0000,&h0F1A,&h6F08,&h0479
137:  Data &h03E8,&h0000,&h0F1A,&h558F,&h4879,&h0000
138:  Data &h0F18,&h4879,&h0000,&h0F18,&h4879,&h0000
139:  Data &h0F18,&h4879,&h0000,&h0F18,&h6100,&hFC52
140:  Data &h4FEF,&h0010,&h33DF,&h0000,&h0F1C,&h3F39
141:  Data &h0000,&h0F1C,&h4879,&h0000,&h0F1E,&h6100
142:  Data &hFE54,&h5C8F,&h3F39,&h0000,&h0F1C,&h3F3C
143:  Data &h0014,&h4879,&h0000,&h0F18,&h4879,&h0000
144:  Data &h0F18,&h4879,&h0000,&h0F18,&h4879,&h0000
145:  Data &h0F18,&h6100,&hFD66,&h4FEF,&h0014,&h558F
146:  Data &h3F39,&h0000,&h0F1C,&h3F3C,&h0003,&h6100
147:  Data &hFD1A,&h588F,&h33DF,&h0000,&h0F18,&h49F9
148:  Data &h0000,&h07F2,&h47F9,&h0000,&h0F32,&h7A07
149:  Data &h16DC,&h57CD,&hFFFC,&h3F39,&h0000,&h0F1C
150:  Data &h4267,&h4879,&h0000,&h0F3E,&h6100,&hFD82
```

```
151:  Data &h508F,&h4279,&h0000,&h0F18,&h3F39,&h0000
152:  Data &h0F1C,&h3F3C,&h010E,&h3F39,&h0000,&h0F18
153:  Data &h3F3C,&h000A,&h4879,&h0000,&h0F32,&h6100
154:  Data &hFE5A,&h4FEF,&h000C,&h0C79,&h00C6,&h0000
155:  Data &h0F18,&h6C08,&h5679,&h0000,&h0F18,&h60CC
156:  Data &h558F,&h3F39,&h0000,&h0F1C,&h3F3C,&h0001
157:  Data &h6100,&hFCA0,&h588F,&h33DF,&h0000,&h0F18
158:  Data &h3F39,&h0000,&h0F1C,&h3F3C,&h010E,&h3F3C
159:  Data &h00C8,&h3F3C,&h000A,&h4879,&h0000,&h0F32
160:  Data &h6100,&hFE10,&h4FEF,&h000C,&h558F,&h3F39
161:  Data &h0000,&h0F1C,&h3F3C,&h0003,&h6100,&hFC68
162:  Data &h588F,&h33DF,&h0000,&h0F18,&h13FC,&h0020
163:  Data &h0000,&h0F37,&h13FC,&h0021,&h0000,&h0F38
164:  Data &h33FC,&h0190,&h0000,&h0F18,&h3F39,&h0000
165:  Data &h0F1C,&h3F3C,&h015E,&h3F39,&h0000,&h0F18
166:  Data &h3F3C,&h000A,&h4879,&h0000,&h0F32,&h6100
167:  Data &hFDBE,&h4FEF,&h000C,&h203C,&h0000,&h00CA
168:  Data &hB079,&h0000,&h0F18,&h6C08,&h5779,&h0000
169:  Data &h0F18,&h60C8,&h558F,&h3F39,&h0000,&h0F1C
170:  Data &h3F3C,&h0001,&h6100,&hFC00,&h588F,&h33DF
171:  Data &h0000,&h0F18,&h3F39,&h0000,&h0F1C,&h3F3C
172:  Data &h015E,&h3F3C,&h00C8,&h3F3C,&h000A,&h4879
173:  Data &h0000,&h0F32,&h6100,&hFD70,&h4FEF,&h000C
174:  Data &h3F39,&h0000,&h0F1C,&h3F3C,&h0001,&h4879
175:  Data &h0000,&h0F3E,&h6100,&hFC5A,&h508F,&h558F
176:  Data &h3F39,&h0000,&h0F1C,&h3F39,&h0000,&h0F20
177:  Data &h6100,&hFBB0,&h588F,&h33DF,&h0000,&h0F18
178:  Data &h3F39,&h0000,&h0F1C,&h3F39,&h0000,&h0F2C
179:  Data &h4879,&h0000,&h0F18,&h4879,&h0000,&h0F18
180:  Data &h4879,&h0000,&h0F18,&h4879,&h0000,&h0F18
181:  Data &h6100,&hFBB8,&h4FEF,&h0014,&h7A01,&h23C5
182:  Data &h0000,&h0F14,&h52B9,&h0000,&h0F14,&h0CB9
183:  Data &h0004,&hE201,&h0000,&h0F14,&h66EE,&h6000
184:  Data &hFDB2,&h4E5E,&h4E75,&h0000,&h0000,&h4150
185:  Data &h5249,&h4C2C,&h2000,&h0000,&h0002,&h102A
186:  Data &h8428,&h0C04,&h0612,&h0812,&h0A0A,&h0A0A
187:  Data &h0A1A,&h060A,&h0C0C,&h0C0C,&h0E1A,&h1E1A
188:  Data &h0A1E,&h261A,&h1208,&h0808,&h080A,&h0A08
189:  Data &h0608,&h0608,&h0608,&h0608,&h0604,&h0604
190:  Data &h0604,&h0604,&h0608,&h080C,&h160A,&h0610
191:  Data &h0816,&h0A0A,&h0A0A,&h0A20,&h0C06,&h060C
192:  Data &h080C,&h0808,&h1C0A,&h0606,&h0806,&h1208
193:  Data &h0A06,&h0A06,&h1208,&h1008,&h0606,&h0814
194:  Data &h1008,&h0A10,&h0C0C,&h0806,&h0A10,&h1406
195:  Data &h100A,&h0806,&h0606,&h0E06,&h060C,&h0A06
196:  Data &h0606,&h1010,&h0606,&h0E08,&h0C06,&h0A0A
197:  Data &h1008,&h0A10,&h0612,&h1010,&h0808,&h0806
198:  Data &h0A0A,&h1408,&h0A10,&h0612,&h0E0A,&h0E06
199:  Data &h0C06,&h0606,&h0606,&h0610,&h060A,&h0000
```

Listing und Artikel Ende



Genug davon gehört: DTP, WYSIWYG, Setzerei auf dem Schreibtisch, Drucksachen mit dem Computer erstellen, und, und, ... Jetzt heißt's selber loslegen: Desktop Publishing für alle! Ab sofort gibt's den Publisher für alle ATARI ST, der hält, was DTP verspricht: BECKERpage ST.
Sie wollen es nicht glauben? Sehen Sie selbst: diese Anzeige wurde komplett mit BECKERpage ST realisiert.
BECKERpage ST hat nicht nur alleTextverarbeitungsfunktionen drauf, schafft nicht nur verschiedene Schriftarten wie Helvetica, Times und Courier in allen Variationen (fett, kursiv, negativ, outline, gesperrt), kann nicht nur eigene Schriftbilder entwerfen (mit dem integrierten Schrifteditor FOMA) - nein, BECKERpage ermöglicht auch die Kreation und Integration beliebiger Grafiken.Die Illustration dieser Anzeige wurde mit dem mitgelieferten Grafikprogramm PROFI PAINTER erstellt. Ebensogut können Grafiken und Bildschirmausschnitte aus anderen Programmen übernommen und beliebig variiert werden. Dafür sorgt die integrierte Snap-Shot-Accessory PAINTER FOTO (siehe Hardcopy links in Anzeige).



Nicht genug. BECKERpage ST ermöglicht vollständiges Seitenlayout am Bildschirm. Natürlich mit der Maus: mehrspaltiger Seitenumbruch, Blindtextfunktion, Picture Wrap, beliebig abgestufte Raster, Kreise, Linien, Rechtecke, und, und ...
Genug gehört? Dann ran an DTP! BECKERpage ST macht's möglich.
Minimalkonfiguration: ATARI ST mit ROM TOS und 1 MByte Speicher, ATARI-Laserdrucker oder NEC P6, P7, P 2200, Star NB 15, Epson 9- und 24-Nadeldrucker und Kompatible.

BECKERpage ST

Version 1.0,
inkl. PROFI PAINTER nur DM **398,-**

COUPON

Hiermit bestelle ich

Name, Vorname

Straße, Ort

DATA BECKER
Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010

TEST

PROSPERO-FORTRAN IM DETAIL

Dem Compiler auf die Finger geschaut

In der Februar-Ausgabe wurde über die neueste Version von Prospero-Fortran berichtet, die einerseits über ihre Workbench besser in GEM eingebunden ist, andererseits aber auch durch eine umfangreiche Unterprogrammbibliothek dem Programmierer einen erweiterten Zugriff auf die leistungsfähigen GEM-Routinen ermöglicht.

Hilfeoption des Debuggers

```
>>
>>
>>
>>
>>
>>H
PROBE checks for breakpoints,  tracing,  watching, or striking of any key at
the  start of each line of the source program on which a  statement  starts.
When  execution of the program is interrupted,  the following  commands  are
available:

A(ssign) new value to variable     O(utput) to alternate screen
B(reak) into program at event      P(rofile) on/off
C(alls) to reach current point     QUIT to operating system
D(isplay) value of variable(s)     R(oute) taken to current point
E(cho) screen output to file       S(tep) execution through source
G(o) on with execution             T(race) execution in object code
H(elp) explain command use         V(iew) alternate output screen
K(ey) interruption on/off          W(atch) variable or source lines
L(ist) source lines                X (eXecute) program

     Further help is available on each of these.

      press RETURN to continue; SPACE to leave HELP . . .
```

In diesem Artikel sollen einige Aspekte dieses Compilers im Detail betrachtet werden. Dabei haben wir vor allem diejenigen Programmierer vor Augen, die zwar schon Erfahrung mit Fortran haben, sich bisher aber noch nicht weiter mit GEM beschäftigt haben und auch mit dem neuen Prospero-Fortran noch keinen intensiven Kontakt hatten.
Zunächst werden wir uns mit dem Debugger beschäftigen, anschließend wird ein wenig gefensterlt, wobei uns einige GEM-Routinen assistieren werden, danach kommen einige Vergleichsmessungen mit der alten Fortran-Version und mit Fortran-Versionen auf anderen Rechnern. Zum Schluß geben wir noch Hinweise auf weitere, bisher gefundene Fehler und Unstimmigkeiten.

Fehlersuche mit PROBE

Ein Debugger ist ein Hilfsmittel, im allgemeinen ein Zusatzprogramm, welches bei der Fehlersuche eingesetzt wird. Normalerweise werden allenfalls Laufzeitfehler mit Fehlerart und Programmzeile gemeldet. Dies ist oft aber nur die Wirkung von Fehlern, die Ursache liegt dann meist im Dunkeln. Logische Fehler findet man auf diese Weise ohnehin nicht. Es kommt daher darauf an, den Ablauf eines Programms genau zu beobachten und an manchen Stellen den Wert signifikanter Variablen zu prüfen. Hierzu quält man sich oft mit zusätzlich ins Programm aufgenommenen Druckanwei-sungen: “Hier ist Unterprogramm ABC”, “Die Variable XYZ hat den Wert ... “ und ähnliches. Diese Methode ist zwar jedem gestandenen Fortran-Programmierer wohlvertraut, nichtsdestotrotz aber sehr umständlich: Die zunächst als fehlerhaft verdächtigten Stellen sind natürlich völlig in Ordnung, man muß ganz woanders suchen, also neue Druckbefehle ‘rein, nochmal übersetzen, binden, wieder falsch, ... Wir kennen das!
Mit einem Debugger hingegen ist man viel besser dran: Das Originalprogramm wird unter der Kontrolle dieses Hilfsprogramms ausgeführt, man kann sich zur Laufzeit überlegen, was man ansehen möchte, wo man weitermachen will und was einem sonst noch so einfällt. Debugger sind natürlich nicht genormt, und so ist ihre Bedienung und Leistungsfähigkeit recht unterschiedlich, wenn man überhaupt einen hat. Selbst auf Großrechnern ist dieses wichtige Hilfsmittel nicht immer selbstverständlich.

Der Einstieg

Soll eine Programmeinheit mit PROBE untersucht werden, so ist sie mit der Option ‘N’ zu übersetzen, alle anderen Programmteile und Programme sind für den Debugger transparent. Beim Übersetzen mit ‘N’ werden die Hilfsdateien der Typen ‘.NAM’ und ‘.SYM’ erzeugt, die PROBE bei seiner Arbeit benötigt. Verwendet man zusätzlich noch die ‘L’-Option, dann kann PROBE die dadurch erzeugte Auflistung des Quellprogramms in der ‘.PRN’-Datei während einer Sitzung ebenfalls verwenden. PROBE kann direkt vom Desktop gestartet oder aus der Workbench über ‘Debug Program’ oder die Tastenkombination (Alternate+P) aufgerufen werden. Im folgenden werden wir den letzteren Fall betrachten.

geht weiter...

Nach (Alternate+P) darf man zunächst anhand der üblichen Auswahlbox das gewünschte Programm aussuchen. Leider gibt es keine direkte Möglichkeit, das gerade entwickelte Programm einzustellen. Oder ist die nur nicht gut genug dokumentiert? Die beim Aufruf von PROBE aus der Workbench verwendeten Pfadnamen sind etwas verwirrend: PROBE.PRG wird im Pfad für "compiler overlays" gesucht; für das zu testende Programm wird in der Auswahlbox der Pfad für "user files" voreingestellt; die Dateien vom Typ '.PRN' und '.NAM' werden im Pfad für "libraries" gesucht, wo auch die Protokolldatei PROBE.LOG abgelegt und die Datei PROBE.HLP mit den Hilfstexten vermutet wird. Wird in der laufenden Sitzung für eine nicht gefundene Datei einmal ein anderer Pfadname eingegeben, so gilt der dann aber auch für die nächste gesuchte Datei.

Ein Beispiel

Natürlich können, dürfen und wollen wir hier nicht die Lektüre des Handbuchs ersparen. Wir geben aber einige Tips und Hinweise, die den Einstieg erleichtern und einen kleinen Eindruck von der Leistungsfähigkeit dieses Debuggers geben. Ganz instruktiv ist hierbei der Ausdruck in Bild 1. Dies ist das Protokoll einer Sitzung, die mit ECHO in die Datei PROBE.LOG protokolliert und dann mit einem Editor nachbearbeitet wurde: Alle Benutzereingaben sind unterstrichen. Das Protokoll enthält stets nur die von PROBE erzeugte, nicht die vom getesteten Programm erzeugte Ausgabe und auch nicht die vom Benutzer eingegebenen Pfadnamen. Diese wurden in Bild 1 nachträglich noch eingesetzt. Die vom Benutzer eingegebenen PROBE-Befehle stehen dagegen auch wieder im Protokoll.

Bild 1 vermittelt zunächst einen treffenden Eindruck von dem Chaos mit den Pfadnamen. Positiv fällt dagegen der Umfang der HELP-Funktion auf, die hier nur andeutungsweise demonstriert wurde. Zu jedem Punkt kann noch ein ausführlicher Hilfstext abgerufen werden. So etwas ist vorbildlich! Mit DISPLAY wird der Inhalt von Variablen und Feldern angezeigt. Natürlich kann man genau angeben, was man sehen möchte; ohne Parameter wirkt der Befehl auf alle Größen der aktuellen Programmeinheit.

```
     .MAIN  (source file: DEMO1)
   'DEMO1.PRN' is not in the default directory.
  Which directory is it in? D:\
     1 *        A = 10.
  >>HELP
   'PROBE.HLP' is not in  D:\.
   Which directory is it in? D:\PROFOR\
   PROBE checks for breakpoints,  tracing,
  watching, or striking of any key at
  the  start of each line of the source
  program on which a  statement  starts.
   When  execution of the program is interrupted,
  the following  commands  are
  available:

 A(ssign) new value to variable
          O(utput) to alternate screen
 B(reak) into program at event
          P(rofile) on/off
 C(alls) to reach current point
          QUIT to operating system
 D(isplay) value of variable(s)
          R(oute) taken to current point
 E(cho) screen output to file
          S(tep) execution through source
 G(o) on with execution
          T(race) execution in object code
 H(elp) explain command use
          V(iew) alternate output screen
 K(ey) interruption on/off
          W(atch) variable or source lines
 L(ist) source lines
          X (eXecute) program


       Further help is available on each of these.

 . . . press RETURN to continue;
           SPACE to leave HELP . . .
 >>DISPLAY
 'DEMO1.NAM' is not in   D:\PROFOR\.
 Which directory is it in? D:\
                 A    (REAL*4  )  0.0000000E+00
                 B    (REAL*4  )  0.0000000E+00
                 C    (REAL*4  )  0.0000000E+00
 >>PROFILE
  profiling enabled.
 >>STEP
    2 *               B = 5.
 >>DISPLAY
                 A    (REAL*4  )  1.0000000E+01
                 B    (REAL*4  )  0.0000000E+00
                 C    (REAL*4  )  0.0000000E+00
 >>BREAK 8
 >>WATCH C
 >>GO
   ADD
   12 *         1        X3 = X1 + X2
                'WATCH C'    1.5000000E+01
   12 *         2        X3 = X1 + X2
                'WATCH C'    1.6500000E+01
   .MAIN
    8 *                  STOP
```

Bild 1: Beispiel einer Sitzung mit PROBE. Alle Benutzereingaben sind unterstrichen. Deutlich zu sehen ist das Problem mit den Pfaden.

Goodies

Nach Eingabe von PROFILE wird gezählt, wie oft jede Programmzeile ausgeführt wird. Gerade bei komplexen Programmen gewinnt man so wichtige Hinweise für die Optimierung. Mit LIST wird diese Liste ausgegeben, zuvor sollte natürlich das Mitschreiben der Ausgabe durch ECHO in die Datei PROBE.LOG eingeschaltet werden. Fragen Sie jetzt aber bitte nicht, in welchem Ordner diese Datei anschließend

```
>>LIST  1..200
    1 *      1          A = 10.
    2 *      1          B = 5.
    3 *      1           CALL ADD (A, B, C)
    4 *      1          PRINT *, C
    5 *      1          B = 6.5
    6 *      1           CALL ADD (A, B, C)
    7 *      1          PRINT *, C
    8 *                 STOP
    9 *                 END
   10 +           *
   11                     SUBROUTINE ADD (X1, X2, X3)
   12 *      2          X3 = X1 + X2
   13 *      2          RETURN
   14 *      2          END
```

Listing 1: Ausgabe der mit dem Beispiel aus Bild 1 erzeugten Statistik. Das Hauptprogramm wurde einmal, das Unterprogramm zweimal ausgeführt. Alle ausführbaren Anweisungen werden mit '' gekennzeichnet.*

gesucht werden muß! Ein weiteres ECHO schaltet die Ausgabe übrigens wieder ab. Die so erzeugte Liste für das Beispiel aus Bild 1 finden Sie in Listing 1.

Ganz wichtig ist auch der Befehl STEP. Ohne Parameter wird genau eine Anweisung ausgeführt und dann die nächste zur Ausführung anstehende Zeile angezeigt. Fehlt die '.PRN'-Datei, dann wird nur die Zeilennummer gemeldet. Und nun raten Sie mal, was passiert, wenn man eine Zahl als Parameter angibt!

Mit 'BREAK 8' wird das Programm vor Ausführung der Anweisung in der Quellzeile Nummer 8 unterbrochen und PROBE meldet sich zurück. Warum hier gerade die Zeile Nummer 8 angegeben wird? Schauen Sie sich einmal die Auflistung des Quellprogramms in Listing 1 an! Irgendwann muß man ja, bevor sich alles in Wohlgefallen auflöst, die mit PROFILE erzeugte Statistik ausgeben. Der Befehl WATCH ist wesentlich leistungsfähiger, als man zunächst glaubt. Man kann beispielsweise, wie hier geschehen, die Veränderung von Variablen oder Feldelementen überwachen. Dies kann auch noch von Bedingungen abhängen, etwa 'WATCH C<=5.0'. Stets wird jedoch nicht einfach nur auf die Verwendung des angegebenen Namens in der aktuellen Programmeinheit geachtet; vielmehr wird jeder Zugriff auf den zugehörigen Speicherplatz auch aus aufgerufenen Unterprogrammen gemeldet, wo die untersuchte Größe natürlich ganz anders heißen kann.

Kür verpatzt

Während einer Sitzung mit PROBE kann durch den Befehl X ein beliebiges anderes Programm gestartet werden. Längere PROBEn könnte man dadurch wesentlich erleichtern: Aus dem Desktop wird zunächst PROBE aufgerufen und dann erst die Workbench mittels des Kommandos 'X F_BENCH'. Während der gesamten Sitzung läuft dann jedes mit der N-Option übersetzte Programm automatisch unter PROBE, ohne daß der Debugger jedesmal erneut aufgerufen und geladen werden müßte. Dies klingt nun zwar alles sehr schön, zuvor sollten die Programmentwickler allerdings ihren Debugger selbst einmal debuggen. Nach dem Aufruf von PROBE und Eingabe des Befehls 'X F_BENCH' - der Cursor ist selbstverständlich nicht zu sehen - wird zunächst einmal der Quellcode eines mysteriösen Programms namens P03MG angemahnt. Dieses Ansinnen kann man nun mittels Return-Taste zurückweisen und die Chose anschließend endlich durch den Befehl GO zum Laufen bringen. Der daraufhin sichtbare Anfangsbildschirm der Workbench sieht allerdings schon recht merkwürdig aus; auch scheinen sich einige Zeiger zu verhaspeln, so daß bei mehrfachem Start eines Programms mit (Alternate+R) dieses an unterschiedlichen Stellen begonnen wird. Wenn aber in Zukunft einmal alles so läuft, wie im Handbuch beschrieben, dann wäre dies gewiß ein enormer Komfort.

Und nun zu den Menüs...

Die für GEM charakteristischen Betriebssystem-Routinen gliedern sich in zwei Hauptgruppen: AES (Application Environment Services) und VDI (Virtual Device Interface). Die VDI-Routinen dienen hauptsächlich der graphischen Ausgabe, während die AES-Routinen vor allem die WIMP-Benutzerschnittstelle verwalten. WIMP steht hier für "windows, icons, mice, pull-down menus". Am

geht weiter...

```
 1:   * PROG1.FOR   02-02-88   VK
 2:    * Demonstration einiger GEM-Funktionen...
 3:    * - Menueleiste einrichten und anzeigen
 4:    * - Tochterprogramme starten
 5:   * - Textbox anzeigen
 6:  *
 7:          PROGRAM PROG1
 8:  *
 9:    * Verwendete Funktionen des Bindings...
10:          INTEGER*4
11:         -       menu_create,
                   !erzeugt Menueleiste ohne RSC-Datei
12:         -        menu_title,          !Menuepunkt anfu"gen
13:         -       menu_item,
                   !Funktion an Menuepunkt anfu"gen
14:         -       AESret,              !Parameter holen
15:         -       form_alert
                   !zeichnet Alarm-(alert-)Box
16:  *
17:   * Eigene Variable und Felder
18:          INTEGER*4
19:         -        MENUE_BAUM,          !Adresse des Menue-Baums
20:         -        MENUE1, MENUE2,      !Titel der Menues 1 und 2
21:         -       FKT1,                 !Funktion von Menue 1
22:         -         FKT2, FKT3, FKT4,   !Funktionen von Menue 2
```

Listing geht weiter...

Listing 2: Das Oberprogramm PROG1. Die GEM-Funktionen sind in den Kommentaren erläutert. (u" = ü, "ü" ist unter dem von Prospero mitgelieferten Editor nicht verfügbar.)

```
23:          -        GEW_MENUE, GEW_FKT,
                  !gewa"hltes Menue und Funktion
24:          -        DUMMY              !Hilfsvariable
25:            INTEGER*2    PUFFER(0:7) !Ergebnisse der Menuewahl
26:            CHARACTER*4 TEXT         !Ausgabe im Alarmfenster
27:    *
28:            INCLUDE 'gemconst'
29:    *
30:    * Hier werden die Menues aufgebaut...
31:    *
32:    *        {Immer zuerst GEM-Anwendung initialisieren}
33:           CALL appl_init
34:           IF (AESret() .lt. 0) GOTO 999
                 !Das ging schief!
35:    *
36:    *       {Menuebaum mit erstem Menue erstellen}
37:           MENUE_BAUM = menu_create( 15, 15,
                 '  Wie immer '//char(0) )
38:    *
39:    *       {Weitere Menues anfu"gen}
40:           MENUE1 = menu_title( MENUE_BAUM,
                 ' Ausgang '//char(0) )
41:           MENUE2 = menu_title( MENUE_BAUM,
                 ' Programme '//char(0) )
42:    *
43:    *       {Funktionen fu"r Menues 1 und 2}
44:           FKT1 = menu_item( MENUE_BAUM, MENUE1,
                 ' Ende'//char(0) )
45:           FKT2 = menu_item( MENUE_BAUM, MENUE2,
                 ' GEM-Meldung '//char(0) )
46:           FKT3 = menu_item( MENUE_BAUM, MENUE2,
                 '---------'//char(0))
47:           FKT4 = menu_item( MENUE_BAUM, MENUE2,
                 ' TOS-Meldung '//char(0) )
48:    *
49:    *       {Striche schattieren, damit Auswahl verhindern}
50:           CALL menu_ienable( MENUE_BAUM, FKT3, .FALSE. )
51:    *
52:    *       {Menueleiste anzeigen}
53:           CALL menu_bar( MENUE_BAUM, .TRUE. )
54:    *
55:    * Jetzt wird gearbeitet...
56:    *
57:    10     CALL evnt_mesag( PUFFER )
           !Menue anbieten, auf Auswahl warten
58:           GEW_MENUE = PUFFER(3)
                !Gewa"hltes Menue und...
59:          GEW_FKT   = PUFFER(4)
                !...daraus gewa"hlte Funktion
60:           IF (PUFFER(0) .EQ. MN_SELECTED) THEN
61:             IF (GEW_MENUE .EQ. 3) THEN
                   !Menue 3 ist immer das Desk-Menue
62:               DUMMY = form_alert( 1, '[1]['//
63:    -                  '     (C) Copyright:       |'//
64:    -                  '   Gibt es fu"r ein so    |'//
65:    -                  'einfaches Programm nicht!]'//
66:    -                  '[Na klar!]'//char(0) )
67:             ELSEIF (GEW_FKT .EQ. FKT1) THEN
68:              GOTO 999
69:             ELSEIF (GEW_FKT .EQ. FKT2) THEN
70:               DUMMY = form_alert( 1, '[1]['//
71:    -                  'Gleich wird sich ein|'//
72:    -                  'GEM-Programm|'//
73:    -                  'bei Ihnen melden.]'//
74:    -                  '[Na los!]'//char(0) )
75:               CALL EXECPG( '\GEM1.PRG', DUMMY )
                    !Tochterprogramm nachladen
76:              IF (DUMMY .NE. 0) THEN
77:                WRITE (TEXT,'(I4)') DUMMY
78:                DUMMY = form_alert( 1, '[3]['//
79:    -                    'Das GEM-Programm konnte|'//
80:    -                    'nicht gestartet werden.|'//
81:    -                    'Fehlercode: '//TEXT//']'//
82:    -                    '[Ich habe gepfuscht]'//char(0) )
83:              ENDIF
84:             ELSEIF (GEW_FKT .EQ. FKT4) THEN
```

Listing 2 (Fortsetzung)

Listing geht weiter...

interessantesten ist sicher der Einstieg in AES, hier kann nun jeder Fortran-Programmierer seinen Programmen das GEM-typische Aussehen verleihen.

Ohne RSC-Datei

Bisher wurde hierzu immer eine besondere RSC-Datei benötigt, die alle Angaben zu Menues, deren Funktionen und sonstigen Ausgaben enthält und mit einem speziellen Editor erstellt werden muß. Dies kann man zwar mit den Fortran-Programmen auch tun, jetzt gibt es jedoch noch zusätzlich die Möglichkeit, eine Menüleiste durch die Funktion menu_create dynamisch innerhalb eines Programms selbst zu erstellen. Unser Beispielprogramm in Listing 2 macht davon Gebrauch.

Was sind Bindings?

Die AES-Routinen des Betriebssystems werden aus Maschinenprogrammen, die üblicherweise in Assembler geschrieben sind, über einen Software-Interrupt aufgerufen.
Dies ist in Hochsprachen nicht ohne weiteres programmierbar, man benötigt hier einen Satz Unterprogramme, die die entsprechenden AES- (oder auch VDI-) Funktionen aufrufen und dabei auch, falls nötig, das Umsetzen von Parametern besorgen. Diese Unterprogramme sind die sogenannten "Bindings", die man etwa bei jedem C-Compiler für den ATARI mitbekommt. In Prospero-Fortran sind, ab Version 2, solche Bindings nun ebenfalls, wohldokumentiert, enthalten.

Einarbeiten mit Handbuch

Für die AES- und VDI-Bindings von Prospero wird jeweils ein Handbuch mit etwa 250 Seiten Umfang mitgeliefert. Die Qualität dieser Handbücher ist so gut, daß man sich damit auch ohne zusätzliche Literatur in die GEM-Programmierung einarbeiten kann. Zumindest die ersten Gehversuche werden gut unterstützt, wobei allerdings auch die mitgelieferten Beispielprogramme wertvolle Hilfestellung leisten.

```
 85:            DUMMY = form_alert( 1,  '[1][\'//
 86:       -                'Gleich wird ein|'//
 87:       -                'TOS-Programm|'//
 88:       -                'eine Ausgabe abliefern]'//
 89:       -                '[Her damit!]'//char(0) )
 90:            CALL EXECPG( '\TOS1.PRG', DUMMY )
                  !Tochterprogramm nachladen
 91:           IF (DUMMY .NE. 0) THEN
 92:              WRITE (TEXT,'(I4)') DUMMY
 93:              DUMMY = form_alert( 1,  '[3][\'//
 94:       -                 'Das TOS-Programm konnte|'//
 95:       -                 'nicht gestartet werden.|'//
 96:       -                 'Fehlercode: '//TEXT//']'//
 97:       -                 '[Ich habe gepfuscht]'//char(0) )
 98:          ELSE
 99:              CALL menu_bar( MENUE_BAUM, .TRUE. )
                    !Bildschirm aufra"umen
100:          ENDIF
101:        ENDIF
102:   *
103:   *       {'Reverse Video' fu"r gewa"hltes Menue aufheben }
104:          CALL menu_tnormal( MENUE_BAUM, GEW_MENUE,
                 .TRUE. )
105:       ENDIF
106:       GOTO 10
107:   999  CALL appl_exit
108:       END
```

Listing 2 (Ende)

```
  1:       PROGRAM GEM1
  2:       CALL appl_init
  3:        IF (AESret() .lt. 0) GOTO 999
             !Das ging schief!
  4:        DUMMY = form_alert( 1,  '[1][\'//
  5:      -        'Hier ist das|'//
  6:      -        'GEM-Programm|'//
  7:      -        '[Weiss ich!]'//char(0) )
  8:  999   CALL appl_exit
  9:       END
 10:
```

Listing 3: GEM1, ein ordentlicher GEM-Benutzer

```
  1:       PROGRAM     TOS1
  2:       CHARACTER   DUMMY
  3:       PRINT*,
           'Hallo, hier ist TOS1 !                                    '
  4:       PRINT*,
             'Leider ist jetzt das Desktop nicht mehr scho"n.'
  5:       PRINT*,
           'Weiter geht es mit <RETURN>                               '
  6:       READ         (*,'(A)')    DUMMY
  7:       STOP
  8:       END
```

Listing 4: TOS1, ohne alle GEM-Zutaten

Zu den Bindings

Die originalen AES-Routinen geben meist einen Parameter zurück, der oft aber nicht benötigt wird. Viele AES-Funktionen werden daher in Fortran in SUBROUTINE-Aufrufe umgesetzt; wird der Ergebnisparameter doch einmal benötigt, so kann man ihn nachträglich noch mittels der Funktion AESret abholen. Datenaustausch mit den AES-Routinen des Betriebssystems geschieht über Speicherbereiche, deren Startadressen der jeweiligen Funktion mitgeteilt werden. In den Fortran-Bindings wird dies über benannte COMMON-Bereiche realisiert, deren Namen alle mit den vier Zeichen "AES_" beginnen. Ähnlich ist es bei den VDI-Bindings. In den Bindings wird bei Bedarf noch eine Umwandlung von INTEGER*4-Konstanten (aus dem rufenden Programm) in INTEGER*2-Werte (zum Betriebssystem) vorgenommen. Einige für den Aufruf der GEM-Bindings nützliche Konstanten werden durch PARAMETER- und Typ-Anweisungen in der mitgelieferten Datei GEMCONST.FOR vereinbart, die der Programmierer mittels INCLUDE-Anweisung ins Quellprogramm integrieren kann. In unserem Beispielprogramm in Listing 3 wird dies zur Erläuterung auch getan, obwohl lediglich die Konstante MN_SELECTED benötigt wird.

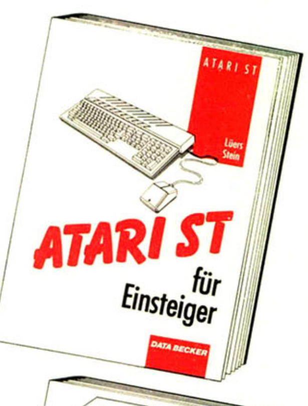
Die Beispielprogramme

Zur Demonstration der Anwendung einiger AES-Funktionen dienen die Programme PROG1, GEM1 und TOS1 aus den Listings 3, 4 und 5. In PROG1 wird ein Menü aufgebaut, mit dem im wesentlichen die Programme GEM1 und TOS1 hinzugeladen und ausgeführt werden können. Dies sind vollkommen selbständige Programme, keine Unterprogramme! In diesem Sinne ist PROG1 eine sogenannte "Shell". Das Programm erklärt sich durch die im Quellcode eingebrachten Kommentare weitgehend selbst. Grundsätzlich wird zu Beginn jedes GEM-Programms mit appl_init Arbeitsspeicher reserviert, der am Ende mit appl_exit wieder freigegeben werden muß. GEM1 zeigt einen kleinen Text unter GEM an und arbeitet mit PROG1 gut zusammen. TOS1 dagegen ist ein reines TOS-Programm und zerstört den Bildschirmaufbau des rufenden GEM-Programms. Um den Effekt vorzuführen, wird dies in PROG1 nicht verhindert. Die Anzahl der Formalparameter der Routine menu_create weicht in den Beispielprogrammen von Prospero vom Handbuch ab; da es mit der Verdoppelung des ersten Parameters (bisher!) keine Fehlfunktionen gab, haben wir uns hier an die Beispielprogramme gehalten.
Bleibt also noch, unseren Lesern viel Spaß beim Fensterln in Fortran zu wünschen. In einer der nächsten Ausgaben werden wir ein größeres Programm vorstellen, bei dem die Möglichkeiten des GEM ausgiebig verwendet und erläutert werden.

Dr. V. Kurz

ENDE

ATARI ST BUCHHITS!



Auspacken und gleich loslegen. Atari ST für Einsteiger macht's möglich. Vom Aufstellen und Anschließen über die Arbeit mit dem GEM-Desktop bis hin zum ST-BASIC – mit diesem Buch haben Sie die optimale Einführung zu Ihrem neuen Rechner. So ist der Erfolg bei Ihrer späteren Arbeit sozusagen schon vorprogrammiert.

ATARI ST für Einsteiger
248 Seiten, DM 29,–



Das meistverkaufte Buch zu GFA-BASIC – und das nicht ohne Grund. Denn hier lernen Sie den kompletten Befehlssatz des GFA-BASIC Version 2.0 und den GFA-Compiler anhand zahlreicher Beispielprogramme kennen. Ganz ohne die sonst übliche, nackte Befehlsübersicht. Das große GFA-BASIC-Buch – natürlich mit einem ausführlichen Einsteigerteil.

Das große GFA-BASIC-Buch
Hardcover, 574 Seiten
DM 49,–



Das Buch zum neuen Super-ST. Einstieg, DTP, Arbeiten mit Laserdrucker und Festplatte, TOS, Blitter, Betriebssystem-Programmierung – was Sie zum MEGA ST wissen müssen, finden Sie hier. Mit kommentiertem TOS-Blitter-Listing. Mit diesem Buch liegt Ihnen die gesamte Mega-Power zu Füßen.

Das große MEGA-ST-Buch
Hardcover, ca. 400 Seiten
inkl. Diskette, DM 69,–

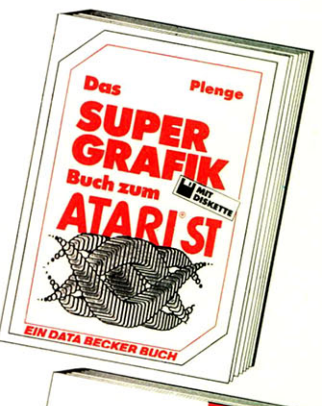


DATA BECKER Führer zum ATARI ST
240 Seiten
DM 29,80



DATA BECKER Führer zu GFA-BASIC
254 Seiten
DM 24,80

Drei Bücher zum ATARI ST ersparen Ihnen das lange Suchen im Handbuch oder in der Fachliteratur. Fehlt Ihnen eine bestimmte Information, können Sie immer ganz gezielt nachschlagen: Die DATA BECKER Führer – die erfolgreichste und kompletteste Serie ihrer Art. Kompetent, zuverlässig und immer griffbereit.



Das Supergrafikbuch zum ST – vollgepackt mit dem Know-how, das jeder engagierte ST-Anwender braucht. Von den Grundlagen bis zu speziellen Problemlösungen wie Programmierung eines Rasterinterrupts oder einer flackerfreien Animation finden Sie hier alles zum Thema Grafik. Mit zahlreichen Utilities in GFA-BASIC, C und Assembler. Ein Buch nicht nur für Grafik-Freaks.

Das Supergrafikbuch zum ATARI ST
Hardcover, 838 Seiten
inkl. Diskette, DM 69,–



Alles zu der Textverarbeitung 1st Word Plus – einschließlich der Zusatzprogramme 1st Mail, 1st Lektor, 1st Proportional und 1st Index. Mit vielen Tips und Lösungen aus der praktischen Arbeit. Nicht nur ein Lehrbuch, sondern auch ein hervorragendes Nachschlagewerk.

Das große Buch zu 1st Word Plus
Hardcover, 288 Seiten
inkl. Diskette, DM 59,–



Intern-Bände von DATA BECKER sind seit jeher Informationspakete ganz besonderer Art. So auch hier: Von der Pinbelegung bis hin zum kommentierten BIOS-Listing wird jedes Detail Ihres Rechners ausführlich beschrieben – natürlich auch der Atari-Blitter. Einfach die Pflichtlektüre für den ST-Profi.

ATARI ST Intern
Hardcover, 637 Seiten
DM 69,–



DATA BECKER Führer zu 1st Word
192 Seiten
DM 24,80

COUPON

HIERMIT BESTELLE ICH

NAME, VORNAME

STRASSE, ORT

zzgl. DM 5,– Versandkosten unabhängig von der bestellten Stückzahl
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei

DATA BECKER
Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010

HD-PACK

Mehr Geschwindigkeit durch Ordnung auf der Festplatte

Welcher Besitzer einer Festplatte kennt nicht das Problem: Nach einiger Zeit scheinen Lese- und Schreiboperationen immer langsamer zu werden. Neben dem psychologischen Phänomen der Gewöhnung des Benutzers an die Geschwindigkeit des neuen Gerätes liegt dies vor allem an der entstandenen Unordnung auf der Platte. Ordnung schafft zum Beispiel HD-Pack, geschrieben in Modula-2.

```
HDPack:
-------

Version 1.2  18.10.1987  Florian Matthes

Buchstabe des logischen Laufwerkes (z.B. 'C') ==>F

Bitte bestätigen Sie die Optimierung für Laufwerk F:
durch die Eingabe des Buchstabens 'P'!

W A R N U N G
Unterbrechen Sie keinesfalls den Programmablauf nach der
Eingabe von 'P', da dies sicher zum Verlust von Dateien und
Directories führen würde!

==>
```

So meldet sich HD-Pack

Dieses Programm reorganisiert die Verteilung der Cluster aller Files und Directories innerhalb einer logischen Partition der Festplatte. Dabei werden Files und Directories in zwei Bereichen der Festplatte dicht an dicht gespeichert, um die Anzahl der Bewegungen der Leseköpfe auf der Festplatte bei der Suche in Inhaltsverzeichnissen und beim Lesen und Schreiben von Files auf ein Minimum zu reduzieren.

Das Ergebnis eines einmaligen Reorganisationslaufes ist eine deutlich erhöhte Geschwindigkeit bei nachfolgenden Lade- und Schreiboperationen für diese Partition. Insbesondere bei I/O-intensiven Programmen (z.B. ST-TeX, Modula- und C-Compilern) kann bei einer vollen Partition eine bis zu 6-fach kürzere Programmlaufzeit erreicht werden.

Bedienung des Programmes

Nach dem Programmstart genügen die Eingabe des Laufwerksbuchstabens der zu optimierenden Partition (z.B. ‘C’) und eine Bestätigung mit der Taste ‘P’, um die Reorganisation zu starten. Die Arbeitszeit des Programmes hängt stark von der Belegung der Platte ab: Bei regelmäßigen Reorganisationsläufen genügen 5 Minuten für eine 16-Megabyte-Partition, während der erste Lauf durchaus auch einmal eine halbe Stunde dauern kann.

Am Ende des Programmes müssen Sie den Rechner mit der RESET-Taste neu booten, damit das GEM-Desktop wieder korrekt initialisiert wird. Ein Abbruch des Programmes während der Programmausführung führt jedoch in 99% der Fälle zu einem Verlust großer Datenmengen in der bearbeiteten Partition.

Außerdem sollten Sie das Programm direkt nach dem Abtippen erst einmal auf einer Partition laufen lassen, von deren Inhalt Sie sich Backup-Kopien erstellt haben. (Falls Sie sich so eine Partition völlig zerstört haben, hilft nur noch die Option ZERO im Programm HDX.PRG.)

Aufräumen tut not

Hier soll nicht zum x-ten Mal das Betriebssystem GEMDOS des ATARI beschrieben werden. Diese Informationen, das Verständnis der verwendeten Optimierungsmethode betreffend, finden Sie u.a. in den am Schluß des Artikel aufgelisteten Grundlagenartikeln. An dieser Stelle soll nur das Prinzip von HD-Pack Thema sein:

GEMDOS, als die oberste Ebene des Betriebssystems TOS, besitzt unter anderem eine Mittlerrolle zwischen dem Anwenderprogramm, das mit Begriffen wie Datei, Directory, Laufwerk arbeitet, und der gerätenahen Ebene des BIOS, das nur Geräte kennt, auf denen jeweils eine Anzahl von Blöcken (genannt Cluster) direkt adressiert werden kann. Jeder Cluster seinerseits umfaßt einen oder mehrere Sektoren, die jeweils eine feste Größe besitzen.

geht weiter...

So ist z.B. die Partition 'C:' für das BIOS ein Gerät mit der Nummer 2, das 16384 Cluster besitzt, von denen jeder 2 Sektoren mit je 512 Bytes umfaßt. Jedes Lesen eines Inhaltsverzeichnisses, einer Datei, das Anlegen und Löschen von Dateien etc. wird von GEMDOS in eine Folge von Lese- und Schreibbefehlen für Sektoren auf einem bestimmten Gerät abgebildet, die an BIOS weitergeleitet werden.

Dabei geht GEMDOS davon aus, daß Cluster aufeinanderfolgender Nummern auch physikalisch (auf der Floppy oder auf der Festplatte) nahe beieinander liegen, so daß Dateien, soweit möglich, nicht über viele weit auseinanderliegende Cluster verstreut werden. Im Laufe der Zeit (nach vielen Schreib- und Löschvorgängen) ist GEMDOS jedoch gezwungen, dies dennoch zu tun. Besonders unangenehm ist das, falls es sich nicht um normale Dateien, sondern um Unterverzeichnisse (Directories) handelt, da diese besonders häufig gelesen werden müssen.

Schiebung auf der Platte

Hier setzt nun HDPack ein: Für eine gegebene Plattenbelegung verschiebt es alle Directories direkt zum 'Anfang' der Festplatte (niedrige Clusternummern), während alle übrigen Dateien zum 'Ende' der Festplatte verschoben werden. Außerdem werden alle Directories so sortiert, daß alle Subdirectory-Einträge vor den restlichen Dateinamen stehen.

Sind im obigen Beispiel 1000 Cluster auf der Platte unbelegt, sowie 50 Cluster für Unterverzeichnisse belegt, ergibt sich die folgende optimierte Belegung:

Cluster	0 bis	49	Directories
Cluster	50 bis	1049	frei
Cluster	1050 bis	16383	übrige Dateien

Die Lücke direkt nach den Directories sorgt dafür, daß GEMDOS auch nach vielen Löschoperationen immer noch schnell freie Cluster finden wird.

Da es nicht möglich ist, zunächst die Festplatte zu leeren, um anschließend die Files in einer besseren Reihenfolge zu speichern, wird die Festplatte durch eine Folge von Clustervertauschungen reorganisiert: Zum Reorganisieren einer Datei (oder eines Directories) werden ihre Cluster nacheinander durchlaufen. Für jeden Cluster wird seine Zielposition (DestCluster) bestimmt. Ist dieser Cluster frei, kann der Clusterinhalt direkt nach DestCluster geschrieben werden. Ansonsten wird der Inhalt des Clusters mit DestCluster vertauscht.

Beispiel: Die erste zu verschiebende Datei befindet sich auf den Clustern 15, 300, 500, 501, 502, 503 und 1052. Sie soll (s.o.) zu den Clustern 1050, 1051 bis 1056 verschoben werden. Dabei sind die Cluster 1052 bis 1054 belegt. Dazu sind folgende Operationen erforderlich:

Schreibe Cluster 15 nach 1050
Schreibe Cluster 300 nach 1051
Tausche Cluster 500 mit 1052
Tausche Cluster 501 mit 1053
Tausche Cluster 502 mit 1054
Schreibe Cluster 503 nach 1055
Schreibe Cluster 500 nach 1056 (!)

Bis auf die letzte Zeile sollten Ihnen diese Operationen klar sein. Jedoch wurde in der letzten Zeile nicht, wie erwartet, Cluster 1052 nach 1056 geschrieben. Das liegt daran, daß bereits im 3. Schritt der Inhalt von Cluster 1052 nach Cluster 500 verschoben wurde.

Programmdetails

Dieser einfache Algorithmus ist der Kern des gesamten Programmes. Die Größe des Programmes resultiert daraus, daß es bei der Verschiebung einer Datei erforderlich ist, Verweise auf den jeweiligen Cluster zu aktualisieren:

1. Die FAT enthält zu jedem belegten Cluster der Platte einen Verweis auf seinen Nachfolger (oder den Wert EOFCluster).
2. Im Inhaltsverzeichnis existiert zu jedem Filenamen ein Verweis (StartCluster) auf den ersten belegten Cluster der Datei.

Neben dieser Buchführungsarbeit werden alle Operationen am Bildschirm protokolliert und Lese- oder Schreibfehler intelligent abgefangen, um Datenverluste zu vermeiden.

Das Hauptprogramm (Listing aufschlagen!) läßt sich in die folgenden Schritte aufteilen: Zunächst wird BIOS gebeten, nähere Details über die Festplattenpartition preiszugeben (BIOS(7) Routine, genannt BIOS. GetBPB).

Hier wird nun überprüft, ob HD-Pack mit diesen Informationen arbeiten kann: Während auf einer Floppy jeder FAT-Eintrag 1.5 Bytes benötigt, sind auf einer Festplatte 2 Bytes erforderlich, um die große Anzahl an Clustern indizieren zu können. Desweiteren wird die Größe von Sektoren, Cluster und Platte in globalen Variablen aufbewahrt.

Nun werden die beiden File Allocation Tables (FAT 1 und 2) gelesen und auf Gleichheit getestet. Nur wenn Ihre Festplatte schweren Schaden genommen hat, werden diese nicht übereinstimmen, so daß Sie vor dem Start von HDPACK.PRG mit dem von ATARI gelieferten Festplatten-Hilfsprogramm HDX.PRG und der Option MARKBAD retten müssen, was noch zu retten ist.

Während des Vertauschungsprozesses werden zwei Informationen für jeden Cluster benötigt: Wo stand dieser Cluster vor dem Beginn des Optimierens? Wohin ist der ursprüngliche Inhalt des Clusters verschoben worden? Diese Daten werden in den Arrays NewPos und OldPos ständig aktualisiert. Dementsprechend werden diese beiden Arrays vor dem Optimieren initialisiert.

Die Prozedur SortDirectory stellt Unterverzeichniseinträge in Verzeichnissen an den Anfang, so daß diese später von GEMDOS schneller gefunden werden können. Diese minimale Optimierung kann bei Bedarf ersatzlos gestrichen werden, da sie nicht Vorbedingung für die nachfolgenden Verschiebungen ist.

Mit MakeNewFAT wird eine FAT erzeugt, in der alle Cluster als frei markiert sind. Wie später auch bei MoveFile werden defekte Sektoren, die in HDX.PRG mit der Option MARKBAD als unbrauchbar gekennzeichnet wurden, ausgelassen.

Anschließend werden mit rekursiven Aufrufen von CompactDirectory die (geschachtelten) Inhaltsverzeichnisse ab dem ersten verfügbaren Cluster auf Platte (FirstCluster) gespeichert. Dabei wird gleichzeitig

die Variable DestCluster erhöht. Nach den Inhaltsverzeichnissen wird eine maximal große Lücke von NumberFree Clustern gelassen. Nach dieser Lücke folgen nun die Dateien, die mit CompactFiles (ebenfalls rekursiv) dicht an dicht gespeichert werden. Am Ende dieser Operation hat DestCluster den letzten adressierbaren Cluster auf der Platte erreicht.
Während NewFAT bereits durch MoveFile aktualisiert wird, müssen die Verweise im Directory auf den Beginn jeder Datei und jedes Unterverzeichnisses explizit mit UpdateLinks (rekursiv) nachgetragen werden.
Schließlich wird NewFAT in zweifacher Ausfertigung zurück auf Platte geschrieben.
Weitere Details finden sich im (hoffentlich) ausreichend dokumentierten Listing, wobei Sie insbesondere auf die Kommentare bei den Variablen und Konstanten achten sollten.

Andere Programmiersprachen

Für Pascal- und C-Programmierer, die ein wenig Verständnis von Modula-2 haben, ist eine Umsetzung in ihre Lieblingssprache mit den obigen Hinweisen und dem Listing sicher nicht schwierig. Für BASIC-Programmierer und absolute Assembler-Fanatiker könnte die Rekursion in CompactDirectory, CompactFiles sowie UpdateLinks problematisch sein. Hier hilft die Standardmethode, lokale Variablen (hier also i, x, r, StartCl) auf einem Stack zu speichern.
Untypischerweise werden in HDPack fast keine Importe aus anderen Modulen vorgenommen, so daß nur die Operationen BIOS(4) (BIOS.RWAbs) und BIOS(7) (BIOS.GetBPB) und der Typ BIOS.BPBPtr (s. ca. Zeile 75 des Listings) extern deklariert werden müssen.

Florian Matthes

Referenzen

Das Betriebssystem des Atari ST
c't 6/86
Das Betriebssystem des Atari ST
c't 10/86 S.122
FAT, Cluster und halbe Bytes
mc 9/86 S. 46
Das ST Betriebssystem (2)
ST 5/86 S. 49
Das ST Betriebssystem (4)
ST 7/86 S. 30
Floppy Spielereien (1)
ST 6/87 S. 20
Floppy-Spielereien (2)
ST 8/87 S. 99


```
  1:  MODULE HDPack;
  2:   (* Optimiere Festplattenbelegung:
  3:       - Sortiere Verzeichnisse so, daß Unterver-
            zeichnisse vor den Files stehen
  4:       - Speichere alle (Unter-)Verzeichnisse ab
            dem 'Anfang' der Festplatte
  5:       - Speichere alle Datenfiles ab dem 'Ende'
           der Festplatte
  6:
  7:       Florian Matthes 18.10.1987   (TDI-Modula
         Version 3.0)
  8:
  9:  *)
 10:  FROM SYSTEM IMPORT ADR, ADDRESS, TSIZE;
 11:  FROM InOut  IMPORT WriteCard, WriteString,
      WriteLn, Write, Read;
 12:  IMPORT BIOS;
 13:
 14:  CONST (* Spezielle Einträge in der FAT:
 15:         FreeCluster   = 0;       (* dieser Cluster
                                       ist unbelegt  *)
 16:         FirstCluster  = 2;      (* Offset zur
                           Umwandlung Cluster-Sektor  *)
 17:         BadCluster    = 0FFF7H; (* dieser
                    Cluster ist defekt                *)
 18:                        (* z.B. Plattenfehler (nicht
                            verwenden)  *)
 19:         EOFCluster    = 0FFF8H; (* Cluster mit
                          diesem oder einem höheren *)
 20:      (* Index sind der letzte Cluster eines    *)
 21:     (* Files                                    *)
 22:         MaxPhysCluster= 03FFFH; (* maximale
                              Clustergröße für HDPack  *)
 23:                (* ==> FAT ist kleiner als 64K *)
 24:         MaxCluster    = 0FFFFH; (* theoretisches
                                  Maximum und auch *)
 25:      (* von ATARI-BIOS tatsächlich erwartetes *)
 26:                                   (* EOFCluster *)
 27:
 28:         MaxDirectoryBlocks = 64; (* beschränkt
                         Sektorgröße in HDPack auf  *)
 29:            (* 64*TSIZE(DirectoryRecord)= 64K *)
 30:
 31:      (* der Inhalt des ersten Buchstabens eines
                             Filenamens in einem      *)
 32:      (* Directory kennzeichnet dessen Typ:      *)
 33:         FreeName      = 0H;       (* Eintrag noch
                                       nie benutzt  *)
 34:         UnusedName    = 0E5H;    (* Eintrag nicht
                                     mehr benutzt *)
 35:         SpecialName   = 02EH;    (* = ORD('.')
                                reservierte Namen: *)
 36:     (* '.'    zeigt auf lfd. Directory *)
 37:     (* '..'   zeigt auf VaterDirectory *)
 38:
 39:  TYPE ClusterIndex = [0..MaxCluster];
 40:         ClusterIndex8086 = CARDINAL;
 41:     (* Wie bei 80xxx Prozessoren: H und L-Byte
                                         vertauscht *)
 42:         attributset =  SET OF (readonlyattr,
 43:                                 hiddenattr,
 44:                                 systemattr,
 45:                                 volumeattr,
 46:                                 subdirattr,
          (* dieses File ist ein Directory *)
 47:                                 archivattr);
 48:        DirectoryRecord     = RECORD
 49:                                Name         :
                               ARRAY[0..10] OF CHAR;
 50:                     (* Filename und Extension *)
 51:                                Attr         :
                              attributset;
 52:                    (* Typ des Files            *)
 53:                                res          :
                               ARRAY[0..13] OF CHAR;
 54:                    (* reserviert               *)
 55:                                  StartCluster:
                              ClusterIndex;
 56:                     (* 1. Cluster des Files   *)
 57:                                Size         :
                                LONG CARD;
 58:                     (* Filegröße in Bytes     *)
 59:                              END;
 60:         DirectoryArray = ARRAY
          [0..MaxDirectoryBlocks-1] OF DirectoryRecord;
 61: (* Struktur eines Sektors in einem Directory *)
 62:
 63:  VAR (* Die 'File Allocation Table' enthält
            folgende Informationen:
 64:            Index 0   Format Version
 65:                  1   unbenutzt
 66:                   2..MaxPhysCluster  Belegung des
                           jeweiligen Clusters
 67:             OrgFAT : FAT vor dem Optimieren
 68:             NewFAT : FAT nach dem Optimieren
 69:       *)
 70:       OrgFAT, NewFAT                : ARRAY
             [0..MaxPhysCluster] OF ClusterIndex8086;
```

geht weiter...

```
 71:      NewPos, OldPos            : ARRAY
             [0..MaxPhysCluster] OF ClusterIndex;
 72:       (* NewPos[i] liefert die momentane
            Position des ehemals i-ten Cluster
 73:        OldPos[i] liefert die ehemalige Position
           des momentan i-ten Cluster
 74:       *)
 75:      BPBPtr                     : BIOS.BPBPtr;
 76:       (* Zeiger auf BIOS Parameterblock für dev.
          Die Deklaration in BIOS ist:
 77:                     POINTER TO RECORD
 78:                        recsiz, clsiz, clsizb,
                           rdlen, fsiz,
 79:                        fatrec, datrec, numcl,
                           bflags: CARDINAL;
 80:                     END;
 81:       *)
 82:      dev                        : CARDINAL;
          (* Gerätenummer 0 = 'A:', 1 ='B:' *)
 83:      (* aus BPBPtr^ abgeleitete Größen: *)
 84:      SectorsPerCluster          : CARDINAL;
 85:      FirstDataSector            : CARDINAL;
 86:       DirectoryRecordsPerSector: CARDINAL;
 87:      FirstDirectorySector       : CARDINAL;
 88:
 89:    PROCEDURE Abort(x: ARRAY OF CHAR; nochanges:
                             BOOLEAN);
 90:    (* Programmabbruch mit Fehlermeldung
 91:    *)
 92:    BEGIN
 93:      WriteLn; WriteString(x); WriteLn;
 94:      IF nochanges THEN
 95:         WriteString('Keine Änderungen auf Platte
                                   durchgeführt.');
 96:      ELSE
 97:         WriteString('Leider sind wahrscheinlich
                   einige Files zerstört worden.');
 98:      END;
 99:      WriteLn;
100:      WriteString('weiter mit einer beliebigen
                              Taste...'); WriteLn;
101:      Read(CH);
102:      HALT;  (* Beende Programmausführung mit
                  Fehlermeldung *)
103:    END Abort;
104:
105:    PROCEDURE SwapLH(x:CARDINAL): CARDINAL;
106:    (* Tausche Low und High-Byte (Umwandlung
         zwischen 80xx - 68000 Format)
107:    *)
108:    BEGIN
109:      RETURN 256 * (x MOD 256) + (x DIV 256)
110:    END SwapLH;
111:
112:    PROCEDURE ClusterToSector(x: ClusterIndex):
          CARDINAL;
113:    (* Berechne den ersten Sektor, in dem der
          Cluster x beginnt
114:    *)
115:    BEGIN
116:      RETURN (x-FirstCluster) *
               SectorsPerCluster + FirstDataSector;
117:    END ClusterToSector;
118:
119:    PROCEDURE SectorToCluster(x: CARDINAL):
                                   ClusterIndex;
120:    (* Berechne den Cluster, in dem sich der
         Sektor x befindet
121:    *)
122:    BEGIN
123:      RETURN (x-FirstDataSector) DIV
                 SectorsPerCluster + FirstCluster;
124:    END SectorToCluster;
125:
126:    PROCEDURE Next(x:ClusterIndex):ClusterIndex;
127:    (* liefere den Nachfolger des Clusters gemäß
          FAT
128:    *)
129:    BEGIN
130:      RETURN SwapLH(OrgFAT[x]);
131:    END Next;
132:
133:    PROCEDURE ReadAbs(buf              : ADDRESS;
                             (* in diesen Puffer *)
134:                     count          : CARDINAL;
                          (* Anzahl der Sektoren *)
135:                     recno          : CARDINAL;
                          (* Index erster Sektor *)
136:                    VAR errorcode: CARDINAL);
                           (* BIOS-Erfolgsmeldung *)
137:    BEGIN
138:       errorcode:= CARDINAL(BIOS.RWAbs(BIOS.Read,
              buf,count,recno,dev));
139:    END ReadAbs;
140:
141:    PROCEDURE WriteAbs(buf             : ADDRESS;
                             (* in diesen Puffer *)
142:                     count          : CARDINAL;
                          (* Anzahl der Sektoren *)
143:                     recno          : CARDINAL;
                          (* Index erster Sektor *)
144:                    VAR errorcode: CARDINAL);
                           (* BIOS-Erfolgsmeldung *)
145:    BEGIN
146:       errorcode:=CARDINAL(BIOS.RWAbs(BIOS.Write,
          buf,count,recno,dev));
147:    END WriteAbs;
148:
149:    PROCEDURE LocateDirectoryRecord(Cluster,
                 Index  : CARDINAL;
150:                   VAR Sector, Offset  :
                       CARDINAL)
                                       : BOOLEAN;
151:    (* Lokalisiere indizierten Directory-Eintrag
         auf der Platte. Beachte
152:      bereits stattgefundene Tauschoperationen.
153:      Das Funktionsergebnis ist FALSE, falls
          dieser Eintrag nicht existiert.
154:      Eingabe:
155:        Cluster: 1.Cluster des zu bearbeitenden
            Directories (Position
156:        vor dem Optimieren). Ist Cluster = 0,
            so handelt es sich
157:        um das Wurzelverzeichnis.
158:        Index  :  Index des gewünschten
                      Eintrages (ab 0 gezählt)
159:      Ausgabe:
160:        Sector :  Sektor, in dem sich der
                      Eintrag auf Platte befindet
161:        Offset :  Index des Eintrages in diesem
                      Sector (ab 0 gezählt)
162:      Beispiel :
163:        LocateDirectoryRecord(1234, 15, sec,
                                  offs)
164:        bestimmt den Sektor, in dem der 16.
            Filenamen des Unterverzeichnisses
165:        steht, das im Cluster 1234 beginnt.
            Innerhalb dieses Sektors (sec) steht
166:        der Filenamen an der Position offs.
167:    *)
168:    BEGIN
169:      IF Cluster = 0 THEN
                   (* Suche im Wurzelverzeichnis *)
170:        Sector:= Index DIV
                   DirectoryRecordsPerSector;
171:        IF Sector+1>BPBPtr^.rdlen THEN
              (* nicht über das Ende hinausgehen *)
172:          RETURN FALSE
173:        END;
174:        INC(Sector, FirstDirectorySector);
175:      ELSE
176:        LOOP
          (* durchlaufe das Directory sektorweise *)
177:          IF Cluster>= EOFCluster THEN
               (* nicht über das Ende hinausgehen *)
178:            RETURN FALSE
179:          END;
180:          IF Index<DirectoryRecordsPerSector*
              SectorsPerCluster THEN
181:            EXIT;
182:          END;
183:          Cluster:= Next(Cluster);
184:          DEC(Index, DirectoryRecordsPerSector*
                             SectorsPerCluster);
185:        END;
186:        Sector:=
             ClusterToSector(NewPos[Cluster]) +
187:           Index DIV DirectoryRecordsPerSector;
188:      END;
189:      Offset:=
           Index MOD DirectoryRecordsPerSector;
190:      RETURN TRUE;
```

geht weiter...

```
191:      END LocateDirectoryRecord;
192:
193:      PROCEDURE GetDirectoryRecord(Cluster:
                   CARDINAL;
194:                 Index   : CARDINAL;
195:                 VAR r   : DirectoryRecord):
                             BOOLEAN;
196:    (* Hole Eintrag r mit diesem Index (0...).
197:       Ergebnis = FALSE, falls dieser Eintrag
        nicht existiert.
198:    *)
199:       VAR Sector, Offset : CARDINAL;
200:           DirectorySector: DirectoryArray;
201:           errorcode      : CARDINAL;
202:    BEGIN
203:       IF LocateDirectoryRecord(Cluster, Index,
          Sector, Offset) THEN
204:         ReadAbs(ADR(DirectorySector),1,Sector,
                                    errorcode);
205:         r:= DirectorySector[Offset];
206:         RETURN (errorcode=0) AND
           (ORD(r.Name[0])# FreeName);
207:      ELSE
208:        RETURN FALSE;
209:      END;
210:    END GetDirectoryRecord;
211:
212:     PROCEDURE PutDirectoryRecord(Cluster:
                                  CARDINAL;
213:                              Index   :
                                  CARDINAL;
214:                              r       :
                      DirectoryRecord): BOOLEAN;
215:    (* Schreibe Eintrag mit diesem Index (0...).
216:       Ergebnis = FALSE, falls dieser Eintrag
       nicht existiert.
217:    *)
218:       VAR Sector, Offset : CARDINAL;
219:           DirectorySector: DirectoryArray;
220:           errorcode      : CARDINAL;
221:    BEGIN
222:      IF LocateDirectoryRecord(Cluster, Index,
        Sector, Offset) THEN
223:         ReadAbs(ADR(DirectorySector),1,Sector,
          errorcode);
224:        IF errorcode#0 THEN
225:          RETURN FALSE;
226:        END;
227:         DirectorySector[Offset]:= r;
228:          WriteAbs(ADR(DirectorySector),1,Sector,
          errorcode);
229:        RETURN errorcode=0;
230:      ELSE
231:        RETURN FALSE;
232:      END;
233:    END PutDirectoryRecord;
234:
235:     PROCEDURE SortDirectory(StartCl: CARDINAL);
236:    (* Sortiere (Unter-)Verzeichnisse so, daß
        Unterverzeichnisnamen vor
237:        den 'normalen' Filenamen auftreten: Dabei
        ist kein Update der FAT
238:        nötig, da alle Vertauschungen innerhalb
        des Files stattfinden.
239:        Sortieralgorithmus: Direktes Einfügen
240:    *)
241:      VAR r,r2    : DirectoryRecord;
242:          i,j,x,y: CARDINAL;
243:    BEGIN
244:     (* Zunächst rekursiv alle Söhne sortieren:*)
245:      i:= 0;
246:       WHILE GetDirectoryRecord(StartCl,i,r) DO
247:        WITH r DO
248:           x:= ORD(Name[0]);
249:           IF (x <> UnusedName) AND (x <> Special
          Name) AND
250:             (subdirattr IN Attr) THEN
251:             SortDirectory(SwapLH(StartCluster));
252:          END;
253:        END;
254:        INC(i);
255:      END; (* WHILE *)
256:
257:       (* jetzt eigenes Directory sortieren: *)
258:      i:= 0;
259:       WHILE GetDirectoryRecord(StartCl,i,r) DO
260:         WITH r DO
261:           x:= ORD(Name[0]);
262:           IF (x <> UnusedName) AND (x <> Special
          Name) AND
263:             NOT(subdirattr IN Attr) THEN
264:             (* i zeigt auf den ersten Eintrag im
                Directory, der keinen
265:                Unterverzeichnisnamen enthält
266:             *)
267:             j:= i+1;
268:             LOOP
269:               IF NOT GetDirectoryRecord(StartCl,
              j,r2) THEN
270:                 RETURN;
271:                 (* Sortierung beendet, da kein
                     Unterverzeichniseintrag
272:                     mehr gefunden wurde.
273:                 *)
274:               END;
275:               y:= ORD(r2.Name[0]);
276:               IF (y <> UnusedName) AND (y <>
              SpecialName)
277:               AND (subdirattr IN r2.Attr) THEN
278:                 (* j zeigt auf den ersten
                     Unterverzeichnisnamen nach i
279:                 *)
280:                 WriteString('Tausche
                            Directory-Einträge: ');
281:                 WriteString(Name); WriteString
                                        ('mit ');
282:                 WriteString(r2.Name); WriteLn;
283:                 IF NOT PutDirectoryRecord
                 (StartCl, j, r) THEN
284:                   WriteString(r.Name);
285:                   WriteString(" kann nicht
                   geschrieben werden ");
286:                   WriteLn; RETURN;
287:                 ELSIF NOT PutDirectoryRecord
                 (StartCl, i, r2) THEN
288:                   WriteString(r2.Name);
289:                   WriteString(" kann nicht
                   geschrieben werden ");
290:                   WriteLn; RETURN;
291:                 END;
292:                 EXIT;
293:               END;
294:               INC(j);
295:             END;
296:           END; (* IF *)
297:         END; (* WITH *)
298:         INC(i);
299:       END; (* WHILE *)
300:     END SortDirectory;
301:
302:     PROCEDURE CountFree(): CARDINAL;
303:     (* zähle die unbelegten Cluster auf der
         Platte
304:     *)
305:       VAR i,Free: CARDINAL;
306:     BEGIN
307:       Free:= 0;
308:       FOR i:= FirstCluster TO BPBPtr^.numcl-1 DO
309:         IF OrgFAT[i]=FreeCluster THEN INC(Free);
          END;
310:       END;
311:       RETURN Free;
312:     END CountFree;
313:
314:     (* Die folgenden Variablen werden global von
         der Prozedur MoveFile
315:        verändert:
316:     *)
317:     VAR DestCluster           : ClusterIndex;
               (* Ziel für nächsten Cluster      *)
318:         FirstFreeDestCluster: ClusterIndex;
               (* Ziel für ersten Datencluster *)
319:
320:      PROCEDURE MoveFile(Cluster: CARDINAL);
321:     (* Tausche alle Cluster des Files mit den
         Clustern ab DestCluster,
322:         Als Seiteneffekt wird DestCluster erhöht
         und Cluster in NewFAT
323:         verkettet.
324:     *)
325:       TYPE Operation = (wr,rd);
326:
```

geht weiter...

```
327:        VAR SourceCluster: ClusterIndex;
              (* tatsächliche Position für Cluster *)
328:          OldDest        : ClusterIndex;
              (* ehemalige Position für DestCluster*)
329:          Predecessor    : ClusterIndex;
              (* zuletzt geschriebener Cluster     *)
330:          Current        : ClusterIndex;
              (* ehemalige Position lfd. Cluster   *)
331:          A,B            : DirectoryArray;
              (* Puffer für je einen Cluster       *)
332:
333:        PROCEDURE OK(Op   : Operation;
334:                 VAR X: DirectoryArray;
335:                     Cl: ClusterIndex):
                         BOOLEAN;
336:          VAR err: CARDINAL;
337:
338:        BEGIN
339:          IF Op = rd THEN
340:            ReadAbs(ADR(X), SectorsPerCluster,
              ClusterToSector(Cl), err);
341:            IF err#0 THEN
342:              WriteString('Warnung: Fehler beim
                         Lesen von Cluster ');
343:              WriteCard(Cl,1);
344:              WriteString(' (Cluster nicht
                              verschoben) ');
345:            END;
346:          ELSE
347:            WriteAbs(ADR(X), SectorsPerCluster,
                      ClusterToSector(Cl), err);
348:            IF err#0 THEN
349:              WriteString('Warnung: Fehler beim
                         Schreiben von Cluster ');
350:              WriteCard(Cl,1);
351:              WriteString
                  (' (Cluster nicht verschoben) ');
352:            END;
353:          END;
354:          RETURN err=0;
355:        END OK;
356:
357:      BEGIN
358:        Predecessor:= 0;
359:        WHILE Cluster<EOFCluster DO
360:          LOOP
361:      (* Lasse defekte Cluster als Ziel aus: *)
362:            WHILE SwapLH(OrgFAT[DestCluster])=
              BadCluster DO
363:              INC(DestCluster)
364:            END;
365:            IF DestCluster<BPBPtr^.numcl THEN EXIT
              END;
366:      (* sollte eigentlich nicht passieren: *)
367:            WriteLn;
368:            WriteString('Warnung: Zu wenige freie
              Cluster ');
369:            WriteLn;
370:            IF FirstFreeDestCluster = 0 THEN
371:              (* vermeide Endlosschleife: *)
372:              Abort('Fataler Fehler: Keine freien
                  Cluster mehr gefunden', FALSE);
373:            ELSE
374:              DestCluster:= FirstFreeDestCluster;
375:              FirstFreeDestCluster:= 0;
376:            END;
377:          END;
378:
379:          (* Tausche jetzt NewPos[Cluster] mit
              DestCluster:                        *)
380:          SourceCluster:= NewPos[Cluster];
381:          OldDest:= OldPos[DestCluster];
382:          Current:= SourceCluster;
383:          (* vorläufig, wird überschrieben, falls
              Block-Austausch erfolgreich *)
384:          IF OrgFAT[OldDest]=FreeCluster THEN
385:            (* Ziel-Cluster ist frei, kann direkt
              überschrieben werden *)
386:            WriteCard(SourceCluster,6);
              WriteString('->');
```

```
387:        IF OK(rd,A,SourceCluster) AND OK(wr,A,
            DestCluster) THEN
388:          NewPos[Cluster]:= DestCluster;
389:          NewPos[OldDest]:= SourceCluster;
390:          OldPos[SourceCluster]:= OldDest;
391:          OldPos[DestCluster]:= Cluster;
392:          Current:= DestCluster;
393:        END;
394:      ELSIF SourceCluster#DestCluster THEN
395:        WriteCard(SourceCluster,6);
            WriteString('<>');
396:        IF OK(rd,A,SourceCluster) THEN
397:          IF OK(rd,B,DestCluster) THEN
398:            IF OK(wr,B,SourceCluster) THEN
399:              IF OK(wr,A,DestCluster) THEN
400:                NewPos[Cluster]:=DestCluster;
401:                NewPos[OldDest]:=SourceCluster;
402:                OldPos[SourceCluster]:=
                                     OldDest;
403:                OldPos[DestCluster]:= Cluster;
404:                Current:= DestCluster;
405:              ELSE
406:                (* mache vorherigen
                    Schreibvorgang rückgängig: *)
407:                IF NOT OK(wr,A,SourceCluster)
                      AND
408:                   NOT OK(wr,A,SourceCluster)
                    THEN (* 2-mal *)
409:                  WriteLn;
410:                  WriteString('Fataler Fehler:
                      Inhalt des Clusters ');
411:                  WriteCard(SourceCluster,4);
412:                  WriteString(' zerstört');
413:                  WriteLn;
414:                END;
415:              END;
416:            END;
417:          END;
418:        END;
419:      END; (* IF unused *)
420:
421:      (* Verkette mit Vorgänger *)
422:      IF Predecessor#0 THEN
423:        IF NewFAT[Predecessor]#FreeCluster
          THEN
424:          WriteLn; WriteString('Warnung:
            Cluster doppelt belegt ');
425:          WriteCard(Predecessor,4); WriteLn;
426:        END;
427:        NewFAT[Predecessor]:= SwapLH(Current);
428:      END;
429:      Predecessor:= Current;
430:      Cluster:= Next(Cluster);
431:      INC(DestCluster);
432:    END; (* WHILE *)
433:
434:    IF Predecessor#0 THEN
435:      IF NewFAT[Predecessor]#FreeCluster
        THEN
436:        WriteLn; WriteString('Warnung:
          Cluster doppelt belegt ');
437:        WriteCard(Predecessor,4); WriteLn;
438:      END;
439:      NewFAT[Predecessor]:=SwapLH(MaxCluster);
440:    END;
441:  END MoveFile;
442:
443:  PROCEDURE CompactDirectory (StartCl:
                CARDINAL);
444:  (* Schreibe Subdirectories adjazent ab
       Cluster 2, aktualisiere NewFAT
445:  *)
446:    VAR i,x: CARDINAL;
447:        r  : DirectoryRecord;
448:
449:  BEGIN
450:    (* Nur falls nicht Wurzelverzeichnis ist
         Verschieben möglich:          *)
451:    IF StartCl#0 THEN MoveFile(StartCl); END;
452:
```

geht weiter...

```
453:      (* Verschiebe jetzt die geschachtelten
            Verzeichnisse:                *)
454:      i:= 0;
455:      WHILE GetDirectoryRecord(StartCl,i,r) DO
456:        WITH r DO
457:          x:= ORD(Name[0]);
458:          IF (x <> UnusedName) AND
459:             (x <> SpecialName) AND
             (* nicht '.' und '..' ! *)
460:             (subdirattr IN Attr) THEN
461:            WriteLn; WriteString(Name);
             Write(':'); WriteLn;
462:            CompactDirectory(SwapLH
             (StartCluster));
463:          END;
464:        END;
465:        INC(i);
466:      END; (* WHILE *)
467:    END CompactDirectory;
468:
469:    PROCEDURE CompactFiles (StartCl: CARDINAL);
470:    (* schreibe Datenfiles adjazent bis zum
          Plattenende, aktualisiere NewFAT
471:    *)
472:      VAR i,x: CARDINAL;
473:          r  : DirectoryRecord;
474:
475:    BEGIN
476:      i:= 0;
477:      WHILE GetDirectoryRecord(StartCl,i,r) DO
478:        WITH r DO
479:          x:= ORD(Name[0]);
480:          IF (x <> UnusedName) AND (x <> Special
          Name) THEN
481:            IF subdirattr IN Attr THEN
             (* rekursiv die Söhne bearbeiten *)
482:              CompactFiles(SwapLH(StartCluster));
483:            ELSE
484:              WriteLn; WriteString(Name);
              Write(':'); WriteLn;
485:              IF Size#0 THEN
486:                MoveFile(SwapLH(StartCluster));
487:              END;
488:            END;
489:          END;
490:        END;
491:        INC(i);
492:      END; (* WHILE *)
493:    END CompactFiles;
494:
495:    PROCEDURE UpdateLinks (StartCl: CARDINAL);
496:    (* ersetzte Eintrag StartCluster für jedes
          File und Unterverzeichnis
497:          durch seinen neuen Wert.
498:    *)
499:      VAR i,x, Start: CARDINAL;
500:          r         : DirectoryRecord;
501:
502:    BEGIN
503:      i:= 0;
504:      WHILE GetDirectoryRecord(StartCl,i,r) DO
505:        WITH r DO
506:          x:= ORD(Name[0]);
507:          Start:= SwapLH(StartCluster);
508:          IF (x <> UnusedName) THEN
          (* auch für '.' und '..' ! *)
509:            IF (subdirattr IN Attr) AND
             (x<>SpecialName) THEN
510:              UpdateLinks(Start);
511:            END;
512:            WriteLn; WriteString(Name);
             Write(':');
513:            IF (Start>=FirstCluster) AND
             (Start<=MaxPhysCluster) THEN
514:              StartCluster:=
               SwapLH(NewPos[Start]);
515:              WriteCard(SwapLH(StartCluster),4);
516:              IF NOT PutDirectoryRecord(StartCl,
             i,r) THEN
517:                WriteLn;
518:                WriteString('Fehler:
                StartCluster für File ');
519:                WriteString(Name);
520:                WriteString(' konnte nicht auf
               den Wert ');
521:                WriteCard(SwapLH(StartCluster),
                 4);
522:                WriteString(' aktualisiert
                werden!'); WriteLn;
523:              END;
524:            END;
525:          END;
526:        END;
527:        INC(i);
528:      END; (* WHILE *)
529:    END UpdateLinks;
530:
531:    PROCEDURE MakeNewFAT;
532:    (* Erzeuge leere NewFAT, in der fehlerhafte
          Cluster bereits markiert sind
533:    *)
534:    BEGIN
535:      NewFAT[0]:= OrgFAT[0];
536:      NewFAT[1]:= OrgFAT[1];
537:      FOR i:= FirstCluster TO BPBPtr^.numcl-1 DO
538:        IF SwapLH(OrgFAT[i])=BadCluster THEN
539:          NewFAT[i]:= SwapLH(BadCluster);
540:        ELSE
541:          NewFAT[i]:= FreeCluster;
542:        END;
543:      END;
544:    END MakeNewFAT;
545:
546:    VAR i          : CARDINAL;
547:        errorcode  : CARDINAL;
548:        CH         : CHAR;
549:        NumberFree: CARDINAL;
550:
551:  BEGIN (* Hauptprogramm *)
552:    Write(CHR(27)); Write('v'); Write(CHR(27));
    Write('E');
553:    WriteString('HDPack:'); WriteLn;
554:    WriteString('——'); WriteLn; WriteLn;
555:    WriteString('Version 1.2  18.10.1987
    Florian Matthes'); WriteLn; WriteLn;
556:    REPEAT
557:      CH:='0'; WriteLn;
558:      WriteString("Buchstabe des logischen
      Laufwerkes (z.B. 'C') ==>");
559:      Read(CH); CH:= CAP(CH); Write(CH);
      WriteLn;
560:    UNTIL (CH>='A') AND (CH<='Z');
561:    dev:= ORD(CH)-ORD("A");
562:
563:    WriteLn;
564:    WriteString('Bitte bestätigen Sie die
    Optimierung für Laufwerk ');
565:    Write(CHR(dev+ORD('A'))); WriteString(':  ');
    WriteLn;
566:    WriteString("durch die Eingabe des
    Buchstabens 'P'!"); WriteLn;
567:    WriteLn; WriteString('W A R N U N G');
    WriteLn;
568:    WriteString("Unterbrechen Sie keinesfalls
    den Programmablauf nach der");
569:    WriteLn;
570:    WriteString("Eingabe von 'P', da dies sicher
    zum Verlust von Dateien und ");
571:    WriteLn;
572:    WriteString('Directories führen würde!');
    WriteLn; WriteLn;
573:    WriteString('==>'); Read(CH); Write(CH);
    WriteLn;
574:    IF CAP(CH)#'P' THEN
575:      Abort('OK: Programmabbruch durch den
      Benutzer', TRUE);
576:    END;
577:
578:    BPBPtr:= BIOS.GetBPB(dev);
579:    IF ADDRESS(BPBPtr) = ADDRESS(0) THEN
580:      Abort('Parameter Block nicht gefunden
      (Laufwerksbuchstabe prüfen)!', TRUE);
581:    END;
582:    (* bestimme die geräteabhängingen Parameter
        und speichere sie global: *)
583:    WITH BPBPtr^ DO
584:      SectorsPerCluster          := clsiz;
585:      FirstDataSector            := datrec;
586:      DirectoryRecordsPerSector:= recsiz DIV
      TSIZE(DirectoryRecord);
587:      FirstDirectorySector       := BPBPtr^.fatrec
```

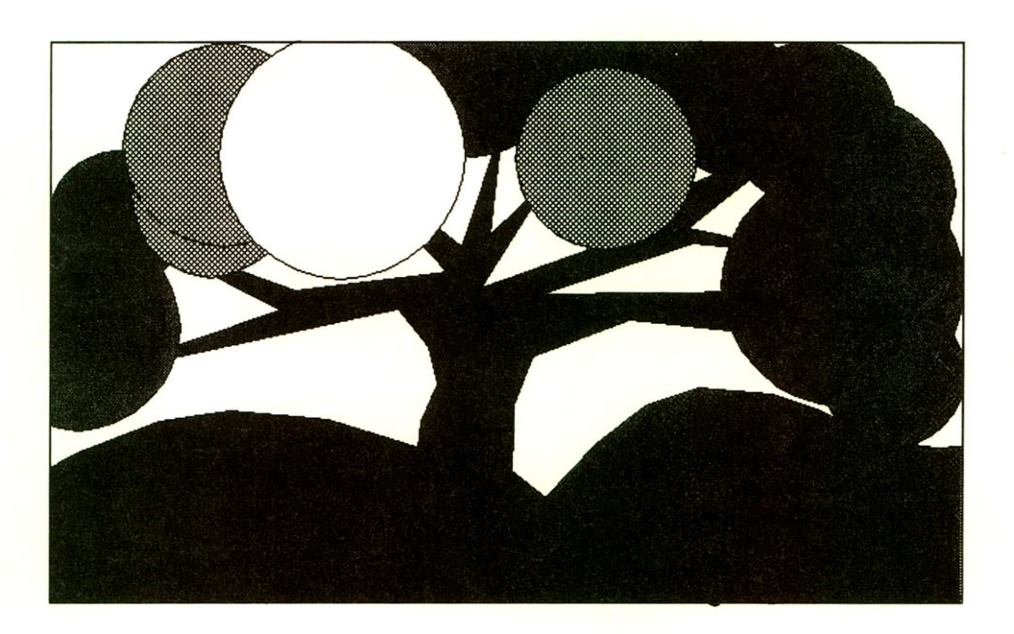
geht weiter...

```
        +BPBPtr^.fsiz;
588:    IF numcl>MaxCluster THEN
589:      Abort('Platte besitzt zu viele (>16384)
        Cluster!', TRUE);
590:    ELSIF numcl<4096 THEN
591:      Abort(
592:       'FAT ist nicht wortweise organisiert.
        HDPack läuft nicht für Floppies',
        TRUE);
593:    ELSIF DirectoryRecordsPerSector>
        MaxDirectoryBlocks THEN
594:      Abort('Sectorgröße auf dem Laufwerk zu
        groß (>65355 Bytes)', TRUE);
595:    END;
596:  END;
597:  WriteString('Parameter Block gelesen...');
      WriteLn;
598:
599:  ReadAbs(ADR(OrgFAT),   (* hole FAT *)
600:        BPBPtr^.fsiz, (* Anzahl Sektoren =
                          Laenge FAT *)
601:        1, errorcode);(* 1. Sektor =
                          1. Sektor FAT   *)
602:  IF errorcode#0 THEN
603:    WriteString('Errorcode =');
        WriteCard(errorcode,4); WriteLn;
604:    Abort('Fehler beim Lesen von FAT 1.',
        TRUE);
605:  END;
606:  WriteString('FAT 1 gelesen...'); WriteLn;
607:
608:  ReadAbs(ADR(OldPos),    (* hole 2.FAT
609:        BPBPtr^.fsiz,  (* Anzahl Sektoren =
                          Laenge FAT *)
610:        BPBPtr^.fatrec, (* 1. Sektor =
                          1. Sektor FAT *)
611:        errorcode);
612:  IF errorcode#0 THEN
613:    WriteString('Errorcode =');
        WriteCard(errorcode,4); WriteLn;
614:    Abort('Fehler beim Lesen von FAT 2.',
        TRUE);
615:  END;
616:  WriteString('FAT 2 gelesen...'); WriteLn;
617:
618:  errorcode:= 0;
619:  FOR i:= 0 TO BPBPtr^.numcl-1 DO
620:    IF OrgFAT[i] # OldPos[i] THEN
621:      INC(errorcode);
622:       WriteCard(i,6); WriteCard(OrgFAT[i],6);
          WriteCard(OldPos[i],6);
623:      WriteLn;
624:    END;
625:  END;
626:  IF errorcode=0 THEN
627:    WriteString('Gleichheit von FAT 1 und FAT
        2 überprüft...'); WriteLn;
628:  ELSE
629:    Abort('FAT1 unterscheidet sich von FAT2 in
        den obigen Clustern', TRUE);
630:  END;
631:
632:  FOR i:= 0 TO BPBPtr^.numcl-1 DO
633:    NewPos[i]:= i; OldPos[i]:= i;
634:  END;
635:
636:  NumberFree:= CountFree();
637:   WriteCard(NumberFree,1);
638:   WriteString(' Cluster noch unbelegt...');
639:  WriteLn;
640:
641:   WriteString('Sortiere Directories...');
      WriteLn;
642:   SortDirectory(0);
643:
644:   WriteLn; WriteString
       ('Erstelle leere FAT...');
645:  MakeNewFAT;
646:  WriteLn;
647:
648:   DestCluster:= FirstCluster;
649:   FirstFreeDestCluster:= 0; (* noch nicht
                               bestimmt *)
650:   CompactDirectory(0);
651:  WriteLn;
652:
653:   FirstFreeDestCluster:= DestCluster;
654:   (* erstes Cluster nach den Directories *)
655:   DestCluster:= FirstFreeDestCluster +
                    NumberFree;
656:   CompactFiles(0);
657:
658:  WriteLn;
659:  WriteString
       ('Aktualisiere Anfangscluster...'); WriteLn;
660:  UpdateLinks(0);
661:  WriteLn;
662:
663:  WriteAbs(ADR(NewFAT),   (* Schreibe FAT *)
664:         BPBPtr^.fsiz,   (* Anzahl Sektoren =
                             Laenge FAT *)
665:         1, errorcode); (* 1. Sektor =
                             1. Sektor FAT   *)
666:  IF errorcode#0 THEN
667:     WriteString('Fehler beim Schreiben von
        FAT 1: Code =');
668:     WriteCard(errorcode,4); WriteLn;
669:  END;
670:   WriteString('FAT 1 geschrieben...');
      WriteLn;
671:
672:  WriteAbs(ADR(NewFAT),   (* schreibe 2.FAT *)
673:         BPBPtr^.fsiz,   (* Anzahl Sektoren =
                             Laenge FAT *)
674:         BPBPtr^.fatrec,(* 1. Sektor =
                             1. Sektor FAT   *)
675:        errorcode);
676:  IF errorcode#0 THEN
677:     WriteString('Fehler beim Schreiben von FAT
          2: Code =');
678:     WriteCard(errorcode,4); WriteLn;
679:  END;
680:   WriteString('FAT 2 geschrieben...');WriteLn;
681:   WriteString('HDPACK erfolgreich beendet.');
      WriteLn; WriteLn;
682:  WriteString('Bitte drücken Sie <RESET> um
      das System erneut zu booten!');
683:  WriteLn;
684:   WriteString('(dies ist zur Initialisierung
      des DESKTOP erforderlich)');
685:  WriteLn;
686:  LOOP END; (* Endlosschleife *)
687: END HDPack.
```

ENDE

ALGORITHMEN & DATENSTRUKTUREN TEIL 6

Schon in der fünften Folge von Algorithmen & Datenstrukturen behandelte ich eine spezielle Baumstruktur, die gegen Degenerierung geschützt ist: den AVL-Baum.
In der heutigen Folge werde ich Ihnen eine ebenso geschützte Struktur vorstellen: den Bayer-Baum oder kurz B-Baum.



Noch'n Baum, wieso?

Alle bisher betrachteten Bäume entstammen der Gattung Binärbäume, da sie maximal zwei Nachfolger je Knoten besitzen. Was von der Struktur her sehr schön, weil einfach ist, hat einen großen Nachteil:
Wird wegen hohem Datenaufkommen eine Baumrepräsentation auf Massenspeichern gewählt, so sind Binärbäume in hohem Maße unefffektiv, da mit jedem Dateizugriff nur jeweils ein Knoten mit einem Schlüsselelement in den Hauptspeichergeladen wird.
Abhilfe kann man sich verschaffen, indem man mehr als einen Schlüssel in jeden Knoten aufnimmt.

Bezeichnung: Der Baum wird dann von einem **Binärbaum** zu einem**Multiwegbaum.**
Die **Knoten** werden als **Seiten** bezeichnet.
Als **Blattseiten** bezeichnen wir die Seiten, die keine Nachfolger besitzen.
Die erste und oberste aller Seiten bezeichnen wir als **Wurzelseite.**

Dabei wird davon ausgegangen, daß die Schlüssel auf jeder Seite, entsprechend der Schlüsselrelation angeordnet sind. Ist nun auf der besuchten Seite der erwünschte Schlüssel nicht auffindbar, so kann er, sofern es sich bei der Seite nicht um eine Blattseite handelt, zwischen seinem unmittelbaren Vorgänger und seinem unmittelbaren Nachfolger gesucht werden.
Zwei Spezialfälle gilt es dabei zu beachten:
Zum einen können alle Schlüssel der Seite größer sein, als unser gesuchter Schlüssel. In diesem Fall ist 'links' vom ersten (kleinsten) Schlüssel weiterzusuchen.
Zum anderen können alle Schlüssel kleiner sein als der gesuchte. Hier ist 'rechts' vom letzten (größten) Schlüssel weiterzusuchen.

Einige grundsätzliche Überlegungen

Leicht erkennt man, daß ein Baum mit **m** Schlüsseln maximal **m+1**-Nachfolger besitzen kann. Ebenso leicht ist einzusehen, daß eine gewisse Obergrenze für die Füllung der Seiten vorzugeben ist, da man sonst nicht wüßte, ob man einen zusätzlichen Schlüssel in die Seite aufnehmen soll, oder ihn auf die Baumnachfolger verweist. Eine andere Überlegung führt zu den Definitionen des Herrn **Bayer:**
Unsere Multiwegbäume neigen nämlich dazu, genauso leicht zu degenerieren wie Binärbäume ohne Ausgleichsmechanismen. Deshalb werden, ähnlich wie bei den AVL-Bäumen, einige Limitierungen eingeführt, deren Beachtung die Degene-

rierung von Multiwegbäumen verhindert.

1. Es wird die **Ordnung n** eines Baumes definiert.

2. Sämtliche Seiten mit Ausnahme der Wurzelseite müssen zwischen n und 2n Elementen besitzen.

3. Jede Seite ist Blattseite oder besitzt genau **m+1** Nachfolger, wobei **m** die Anzahl der Schlüsselelemente der Seite ist.

4. Alle Blattseiten liegen auf einer Höhe.

Datenbankcharakter

Um Ihnen heute eine möglichst effektive Struktur an die Hand zu geben, ist es notwendig, auf die Besonderheiten von Datenbanken, die ja wohl das Hauptanwendungsgebiet von Indexstrukturen sind, einzugehen.
Zunächst einmal: Zeigerstrukturen sind heute **out**. An ihre Stelle tritt der direkte Random-Access-Zugriff auf Diskettendatei.
Was also in den letzten Folgen ein **Pointer** war, wird nun zu einer **Recordnummer**, also einer Zahl die den bezeichneten Datensatz in einer Diskettendatei charakterisiert. Eine weitere Besonderheiten ist die strikte Trennung zwischen Daten und Schlüsseln, worunter deren Abspeicherung in unterschiedlichen Dateien zu verstehen ist.

Grundsätzlich zu Random-Access unter Pascal+

Da Random-Access, wie oben gefordert, vom Vater Pascals, Niklaus Wirth, nicht vorgesehen war, weichen wir heute ein wenig vom Standard ab.
Entgegen der normalen Dateibehandlung in Pascal, nach der eine Datei nur streng sequenziell, will sagen: Datensatz für Datensatz, abgearbeitet werden kann, bietet Random-Access, wie schon der Name sagt, wahlfreien Zugriff auf alle Datensätze der bezeichneten Datei.
Dies geschieht unter Pascal+ wie folgt:
Eine Random-Access-Datei wird unter Pascal+ mit der **reset**-Standardprozedur zum Lesen und Schreiben geöffnet.
Den beiden Standardoperationen **get** und **put** wird nun, als zusätzlicher Parameter, eine **Recordnummer** nachgestellt, die den entsprechenden Datensatz kennzeichnet.
Zu beachten ist dabei, daß man bei den beiden Operationen nicht über das Dateiende hinausschießen sollte, also auf nicht vorhandene Elemente zugreift. Lediglich erlaubt ist ein schreibender Zugriff mit **put** auf ein Element, was eine Position außerhalb der Datei liegt, also das Anhängen eines neuen Elementes.
Erschwert wird die Sache ein wenig dadurch, daß es seitens Pascals keine Prozedur gibt, mit der man die Recordanzahl bestimmen kann. (Zumindestens habe ich keine gefunden.)
Ohne Dokumentation gebe ich Ihnen deshalb die Funktion **dateilaenge** (Listing 6b, Zeilen 28-72) an die Hand, mit der man, unter Angabe eines Dateinamens, der übrigens auch Wildcards beinhalten darf, die erforderlichen Daten erhält.
Die Dateilänge wird dabei, per **call-by-reference**, als zweiter Parameter zurückgegeben.
Als Boolschen Rückgabewert der Funktion erhält man ein Flag, was das Vorhandensein der Datei beschreibt.
Dividiert man nun die Dateilänge durch das Format des Datentyps (**sizeof(<Datentyp>)**, erhält man die Anzahl der Datensätze in der bezeichneten Datei.
Wird noch der Recordoffset (Position des ersten Records. In Pascal+ die Position 0) in Random-Access-Dateien beachtet, so ist vom erhaltenen Wert noch eins abzuziehen, um den letzten Datensatz zu markieren.

Oder etwa nicht ?

Eine Kleinigkeit gilt es noch zu beachten: Es besteht nämlich ein Unterschied zwischen der virtuellen Datei (Menge der geschriebenen Daten) und der physikalischen Datei (Daten die auf dem Massenspeicher schon angekommen sind.).
Der Unterschied besteht genau im von Pascal+ benutzten Puffer für Dateizugriffe. Hier können nämlich noch Datensätze 'rumliegen', die noch nicht geschrieben wurden und auf eine größere Sendung warten, um effektiv weggeschrieben zu werden.
Dieser an sich recht löbliche Effektivitätsgrundsatz führt für unsere Belange dazu, daß aus der physikalischen Dateilänge nicht immer die Anzahl der Datensätze berechnet werden kann.
Dieses kleine Problemchen läßt sich auf zwei Arten umgehen: Entweder wird die Diskettendatei vor jeder **dateilaenge**-Ausführung mit close explizit geschlossen, womit wir Pascal den Befehl geben, seinen Dateipuffer zu leeren, oder die Anzahl der Datensätze wird nur bei der Initialisierung 'hart' ermittelt und in einer Variablen abgespeichert.
Im späteren Verlauf muß man dann nur die Variable entsprechend ändern. (Mit letzterer Variante werden wir uns auch rummühen.)
Beim Ausfügen aus Random-Access-Dateien ist zu einem kleinen Trick zu greifen. Das Ausfügen eines einzelnen Datensatzes aus einer Datei ist, wegen den dabei nötigen Datenverschiebungen, sehr uneffektiv. Deshalb nimmt man in sämtliche Datensätze ein zusätzliches Flag auf, um gelöschte Datensätze markieren zu können. Beim Einfügen ist dieses Flag auf **true** zu setzen und beim eigentlichen Löschvorgang auf **false.** Durch einen speziellen Packalgorithmus kann man, nachdem mehrere Einträge gelöscht wurden, die gültigen Daten (flag auf **true**) zusammenfassen und die ungültigen aus der Datei entfernen. Doch dazu später mehr.

Typstruktur

Welche Typen aus den unterschiedlichen obigen Überlegungen resultieren, sehen Sie im Listing 6a.
Da wären zunächst die beiden Typen **key_type** und **data_type**, die unseren normalen Typzerfall in Schlüssel und Daten symbolisieren. Die beiden Typen **page_ptr** und **data_ptr** sind nur noch der Bezeichnung nach als Zeiger zu erkennen. Sie symbolisieren den Zeigercharakter, den man mit Verweisen auf Recordnummern erreicht, auch ohne Zeiger explizit anzugeben.
Der nächste Typ, **base_type,** entspricht den Einträgen der Datenbank. Er beinhaltet sämtliche Typen (key & data), die es diesmal zu verwalten gilt. Weiterhin ist hier noch ein Zeiger next vorgesehen, der es uns, wie

schon bei den Bäumen, erlaubt, Schlüsselgleichheit zu verarbeiten. Zusätzlich ist noch das oben erwähnte Archivierungsflag aufzunehmen.

Die Seiten der Indexdatei, **page_type,** sehen etwas komplizierter aus. Sie beinhalten zunächst auch ein Archivierungsflag. Die nächste Variable, **anz**, gibt den Füllungsgrad der Seite an. **ptr0** ist der äußerste linke Zeiger der besprochenen Baumstruktur. Die Schlüssel (**key**), ihre jeweiligen rechten Nachfolger (**ptr**), und die Zeiger auf die entsprechenden Datensätze in der Datenbank (**record_nr**) sind zusätzlich in einer Eintragsstruktur **item_type** zusammengefaßt. Von diesen 'Einträgen' benötigen wir maximal Baumordnung mal zwei, also **1..nn** Stück. Es bietet sich eine ARRAY- Repräsentation an, die Sie hier auch realisiert finden.

Beide Seitenarten sind nun noch global (!) als **FILE OF ...** zu deklarieren, wobei die unterschiedlichen Intentionen in die Namenswahl eingehen (Listing 6c, Zeile 21+22).

Ein kleines Beispiel für eine B-Baumstruktur zweiten Grades mit Verzeigerung in die Datenbanken finden Sie in Abbildung 6a. Die (*)-Zeiger stellen dabei die Verzeigerung der Indexseiten dar. Die (**)-Zeiger symbolisieren die Zeiger von der Indexdatei zur Datenbank. Und, last but not least, zeigen die (***)-Zeiger die Verknüpfung innerhalb der Datenbank zwecks Organisierung der Elementnachfolgerliste. Wie weiter oben bereits angeklungen ist, benötige ich heute auch einige Konstanten. Eine zusätzliche Präprozessor-Datei für die Konstanten möchte ich Ihnen allerdings ersparen, zumal es mit Pascal+ ab Version 2.0 möglich ist, Konstanten-, Typ- und Operationsdefinitionen wild zu mischen.

Für die B-Baumkonstanten bemühen Sie also jetzt bitte unser heutiges Demoprogramm (Listing 6c, Zeilen 12-15). Die Konstante **n** beinhaltet den Baumgrad; **nn** den doppelten Baumgrad. Als Ersatz für die Konstante **nil** bei Pointern führe ich heute die Konstante **leer** für ein nicht spezifiziertes Random-Access-Element ein. Weiterhin treffe ich die Konvention, daß das B-Baumwurzelelement immer in erster Dateiposition (0) anzutreffen ist, um mir eine zusätzliche Informationsdatei zu ersparen. Daher noch eine kleine Konstante, **wurzel**, zwecks besserer Lesbarkeit.

Die Operationen auf B-Bäumen im einzelnen (Listing 6b, ab Zeile 75):

create_database

Die Indexdatei und die Datenbank werden mit **rewrite** eröffnet, um sie neu zu erzeugen, und dabei gegebenenfalls alte Inhalte zu löschen.

start_database

Zunächst werden die beiden Da-teienderecords ermittelt (**last_index** und **last_data**). Auf das darauffolgende Öffnen der beiden Dateien zum Random-Access-Zugriff (**reset**) erfolgt noch eine Fallunterscheidung, die zwischen einer leeren Datei und einer Datei mit mindestens einem Element unterscheidet. Entsprechend dem Ergebis dieser Fallunterscheidung ist der Anker unseres B-Baumes (**tree**) entweder auf **leer** oder auf **wurzel** zu setzen.

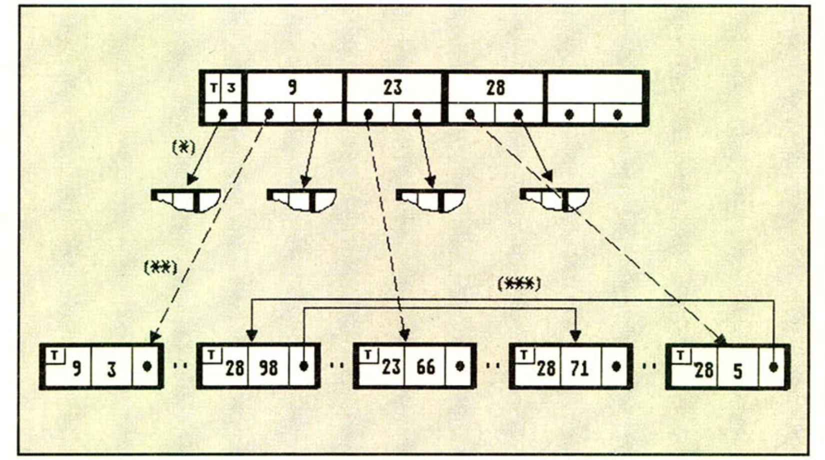


Abbildung 6a: Zeigerstruktur an einem Beispiel 3 Arten von Zeigern: (*) (**) (***)

insert_data

Den Kern von **insert_data** bildet die Prozedur **search**. Mit ihr wird, ähnlich der Suche in Binärbäumen, eine rekursive Suche in den B-Bäumen durchgeführt. Anders als bei den Binärbäumen sind allerdings je Seite maximal **nn** Wegentscheidung zu treffen. Da **nn** frei wählbar und so mitunter recht groß werden kann, bietet es sich an, eine binäre Suche auf dem Schlüsselarray **item** durchzuführen (Zeilen 190-198). Ist das Element gefunden worden (Zeile 199: l-r>1), dann ist am B-Baum keine Änderung vorzunehmen. Das neue Element ist lediglich am Anfang der Nachfolgerliste in die Datenbank einzuhängen (Zeilen 201-206). Andernfalls geht die rekursive Suche weiter (Zeilen 210-213), bis die rekursive Abbruchbedingung (Zeile 175) erfüllt ist.

Da uns die B-Baumdefinition Nummer 4 ein Einfügen unterhalb einer Blattseite verbietet, wird der neue Wert über den Parameter **v** von **search** zurückgegeben. Der Wert **true** des Parameters **h** symbolisiert dabei, daß ein Wert über den Parameter **v** zurückgereicht wurde und in einer höheren Seite einzufügen ist. Dies geschieht durch Check auf h nach dem rekursiven Aufruf von **search** (Zeile 214).

In diesem Fall wird die Prozedur **insert** aufgerufen, die das Einfügen von **v** in die entsprechende Blattseite ausführt und gegebenenfalls, bei wiederholtem Seitenüberlauf, einen neuen Wert für **v** bestimmt.

Im nicht-trivialen Fall des Seitenüberlaufs (Zeilen 137-171) werden dabei einige Eintragsshiftereien vorgenommen, deren genaues Studium ich dem interessierten Leser überlassen möchte, da die Details diesen ohnehin recht großen Beitrag wahrscheinlich sprengen würden.

Im Rumpf der Prozedur **insert_data** wird vor dem Aufruf von **search** zunächst der Datenbankpuffer initialiYsiert und am Ende der Datenbank eingehängt. Dabei ist darauf zu achten, daß zuvor die Datensatzanzahl, **last_data,** inkrementiert wurde.

Nach dem Aufruf von **search** besteht die Möglichkeit, daß sich noch ein Element in **v** befindet (Zeilen 232-250). Dieses Element wird dann das erste der neuen Baumwurzel. Hierbei haben wir auch noch darauf zu achten, daß unsere Vereinbarungen die Wurzel betreffend (Wurzel ist immer in Position 0) eingehalten werden. Die-

se Begebenheit macht normalerweise die Vertauschung von zwei Indexseiten notwendig.
Einen allgemeinen Fall für das Einfügen (und auch Löschen) eines Elementes in einem B-Baum zweiten Grades finden Sie in Abbildung 6b.
Unser Algorithmus versucht das neue Element (91) zunächst in die äußerste linke Seite einzufügen. Da diese aber schon voll ist (Grad * 2 = 4), wird die Seite aufgeteilt. Das mittlere Element der alten Seite (87) wandert dabei über den Parameter v unseres Algorithmus zurück und wird als Entscheidungselement für die beiden neuen Seiten (61 66 / 91 98) in die obere Seite übernommen. Hier wird dadurch wiederum ein Überlauf ausgelöst, der letztendlich zu einer Steigerung der Baumhöhe führt, indem eine neue Wurzelseite (36) angelegt wird.
Anmerkung: Die Random-Access-Struktur ist leider nicht ganz so pflegeleicht, wie unsere bisherigen Zeigerstrukturen. Deshalb muß man beim Arbeiten mit Random-Access sehr darauf bedacht sein, den Inhalt der Puffervariable rechtzeitig (vor eventuellen Änderungen wegzuschreiben (put), um fehlerhafte Variablenbelegungen zu vermeiden. Die entsprechenden Programmstellen wirken dadurch allerdings nicht ganz so elegant, wie Sie das bisher gewohnt waren.

Nicht bei insert, aber in einigen der nun folgenden Operationen wird sogar das Anlegen von Kopien der Puffervariablen notwendig, nämlich genau dann, wenn auf mehrere der Indexseiten zugegriffen werden muß.

delete_data

delete_data besitzt mit der Prozedur **search** einen Programmteil, der sich durch sehr ähnlichen Aufbau die Namensgleichheit mit der Prozedur '**search**' unter **insert_data** verdient. Es wird hier ebenfalls der rekursive Baumdurchlauf betrieben, nur daß bei der Auffindung bzw. Nichtauffindung eines Elementes andere Konsequenzen gezogen werden müssen.

Wird ein Element nicht gefunden (Zeile 400, a=leer), ist der Job für **delete_data** erledigt, was Sie an den durchweg negativen Bekundungen in den Zeilen 402-404 leicht feststellen (false/leer/false).
Andernfalls ist unser Problem nicht so einfach.
Grundsätzlich haben wir beim Ausfügen aus Bayer-Bäumen eine ähnliche Situation wie beim Ausfügen aus den schon hinreichend bekannten Binärbäumen.
Der einfachste Fall, der auftreten kann, ist das Ausfügen aus einer Blattseite (Zeilen 429-433). Hier muß eben nur die Seite entsprechend zusammengerückt werden.
Befindet sich unser auszufügendes Element nicht auf einer Blattseite, so ist es, genau wie bei den Binärbäumen, durch den unmittelbaren Vorgänger oder Nachfolger in der Relation zu ersetzen. Der Abwechslung halber nehmen wir diesmal nicht den maximalen linken Nachfolger des rechten Blattelementes (lexikographischer Nachfolger) sondern den maximalen rechten Nachfolger des linken Blattelementes (lexikographischer Vorgänger). Diesen ermittelt uns die, in ähnlicher Form schon häufig benutzte, Prozedur **del**, die auf rekursiven Wegen die Nachfolgerliste hinabsteigt.
Unten angelangt (q=leer; Bedingung in 400; Ausführung in 387-394) ist der k-te Eintrag des Elementes a durch den letzten Eintrag der ermittelten Blattseite p zu ersetzen. Anders als bei den Binärbäumen brauchen wir uns diesmal nicht um einen eventuellen Nachfolger des letzten Eintrages von p zu kümmern, denn der existiert ja laut B-Baumdefinitionen nicht.

Abbildung 6b: Einfügen/Löschen

Die Krone der Unübersichtlichkeit wird von der Prozedur **underflow** erreicht, die benutzt wird, um Seiten mit weniger als n Blattelementen wieder aufzufüllen bzw. mit anderen Seiten zusammenzulegen.
Sie wird immer dann angewandt, wenn man aus einem rekursiven Aufruf, entweder von **search** oder von **del**, zurückkommt und der Parameter **h** true ist und somit einen Unterlauf in der zurückgelassenen Seite aufweist.
In diesem Fall ist der Prozedur **underflow** die Seite mit Unterlauf, **a**, deren Vorgängerseite, **c**, und die Eintragposition, **s**, mitzuteilen, die die genaue Stelle in **c** markiert, deren Nachfolger **a** ist.
Die Seite **b** stellt nun den anderen direkten Vorgänger bzw. Nachfolger der Stelle **s** in **c** dar.
Je nachdem, ob in **a** und **b** nun noch genügend Knoten vorhanden sind, um zwei Seiten zu füllen, findet entweder ein Ausgleich zwischen den beiden Seiten statt (Zeilen 299-309 und 334-346), oder die Seiten werden zu einer Seite zusammengefaßt (Zeilen 313-320 und 350-357).
Die genaue Verbalisierung dieser im Großen einsichtigen, im Detail aber unübersichtlichen Sache schenke ich mir und Ihnen.
Zurück zum Hauptblock von **delete_data.** Nach dem Aufruf von **search** und den hier erfolgten Mißhandlungen unseres B-Baumes besteht noch die Möglichkeit, daß ein Unterlauf in unserem Wurzelknoten aufgetreten ist. Wie Sie den bisherigen Beispielen sicherlich entnommen haben, ist dies bis zu einem gewissen Grad erlaubt:
Entgegen den Definitionen für eine beliebige Seite muß sich lediglich ein Element im Wurzelknoten befinden.
Wenn es aber auch an diesem einen Element mangelt (Zeile 460), muß etwas unternommen werden:
A: Bei leerem linken Nachfolger ist der Baum völlig entleert, und das ist unserem Programm durch Setzen der Variable **tree** auf **leer** mitzuteilen.

B: Bei einem linken Nachfolger ist dieser zu unserer Wurzelseite zu machen, und die ehemalige Wurzel ist als gelöscht zu kennzeichnen (Setzen des Archivierungsflags).
Beachten Sie dabei bitte auch, daß die beiden Datensätze **q** und **tree** auch physikalisch ausgetauscht werden (Zeilen 470-473).

Abschließend sind noch die Datenseiten des gelöschten Schlüssels aus der Datenbank zu entfernen. Zu diesem Zwecke wird in einer WHILE-Schleife die **del_list** durchlaufen, wobei die entsprechenden Archivierungsflags auf false gesetzt werden.

search_ptr

search_ptr stellt nun die dritte und letzte Variante des Suchens in B-Bäumen dar.
Sie tut weiter nichts, als die Random-Access-Adresse eines durch **key** bezeichneten Datenbankeintrages (nicht Index) zu ermitteln. Der prinzipielle Aufbau ist gleich dem bei **insert_data** und **delete_data**, nur das keine so verwegenen Operationen auf der ermittelten Seite ausgeführt werden.

search_first und search_next

search_first sucht den ersten Datensatz mit bezeichnetem Schlüssel **key** in der Datenbank. Hierzu wird die Funktion search_ptr benutzt. Der Erfolg oder Mißerfolg wird über einen Boolschen Funktionsreturn kundgetan.
Weiterhin ist die globale Variable **next_list** zu erwähnen, die das Auffinden eines dem ersten Datensatz nachfolgenden Elements mit **search_next** erst ermöglicht. Durch wiederholten Aufruf von **search_next** kann so die gesamte Elementnachfolgerliste durchschritten werden.

print_tree

Die Prozedur **print_tree** stellt eine Variante des Präorderdurchlaufs durch eine B-Baumstruktur dar. Zunächst werden dabei sämtliche Schlüsselelemente ausgegeben und hierauffolgend die Liste sämtlicher Nachfolger durchlaufen.
Nebenbei wird dabei auch noch Element für Element die Elementnachfolgerliste in der Datenbank durchlaufen. Zusammen mit einer Baumtiefevariablen, **tiefe**, die bei jedem rekursivem Abstieg inkrementiert und mit den Schlüsselelementen ausgegeben wird, ergeben sich recht anschauliche Baumdarstellungen.
Spielt man ein wenig mit den Baumordnungen herum, kann man mit **print_tree** Informationen darüber gewinnen, wie schnell B-Bäume in Abhängigkeit von ihrer Ordnung wachsen.
In Abbildung 6c erkennen Sie, daß die Baumtiefe schon bei leicht erhöhter Ordnung sehr stark abnimmt. Schlußfolgerungen, die man dadurch über die optimale Ordnung gewinnen kann, sind aber sehr von der Größe der Schlüssel und der Organisation auf Datenträgern abhängig. Sie können aber in den meisten Fällen davon ausgehen, daß eine Seite immer dann recht effektiv geladen wird, wenn das gesamte Seitenformat ungefähr der Verarbeitungsgröße auf dem benutzten Massenspeicher entspricht.

Beispiel: Unser ST hat als Grundgröße für Dateien das Cluster (1 kByte).
Unser Variablenformat (Schlüssel in integer) ergibt sich zu 8+8*nn Byte (kleiner Überschlag der Typensumme des RECORD page_type).
Hieraus würde sich eine 'günstige' Ordnung von etwa 63 ergeben.

Testumgebung

Zur Testumgebung (Listing 6c) ist nicht viel zu sagen. Über einen trivialen Operationsaufruf hinaus ist nur die Routine zur Erzeugung einer 'gewissen' Anzahl von Zufallseinträgen zu nennen, die einem beim Testen viel Tipperei abnimmt.

Packen

Den Packalgorithmus zur Entfernung der leeren Datensätze habe ich in einem gesonderten Programm untergebracht (Listing 6d).
Um zu sehen, wie nötig man das Packen mal wieder gehabt hat, besitzt dieses Programm zunächst zwei kleine Funktionen, die durch einfachen Durchlauf der jeweiligen Datei den prozentualen Anteil der gefüllten Datensätze ermittelt (Zeilen 24-44 für die Indexdatei, Zeilen 46-66 für die Datenbank). Das eigentliche Packen geschieht in mehreren Phasen.

A: abstand_besorgen

Zunächst wird die Anzahl der aufeinanderfolgenden gefüllten und leeren Datensätze der beiden Dateien in den Puffern **p_index** und **p_data** untergebracht. Dabei treffen wir die Konvention, daß immer mit einer Sequenz von gefüllten Datensätzen begonnen wird. In den ungeraden Indizes der Puffer haben wir hiernach die Längen der noch gültigen Sequenzen; in den geraden Indizes die Längen der zu entfernenden Sequenzen.

B: abstand_kumulieren

Durch das Rausschmeißen der leeren Datensätze ändern sich die meisten (fast alle) der **page_ptr** und **data_ptr** in den verbleibenden.
Um die neuen Indizes dieser Datensätze mit möglichst geringem Aufwand berechnen zu können, addieren wir die Anzahl der belegt/unbelegt-Sequenzen in den beiden Puffern auf. Und zwar auf folgende Weise:
Von links beginnend, berechnen wir in den ungeraden Indizes die Gesamtanzahl aller (belegt und unbelegt) vorstehenden Sequenzen. In den geraden Indizes berechnen wir nur die Gesamtanzahl der vorstehenden unbelegten Sequenzen.
Mit **new_index** (bzw. **new_data**) können wir danach sehr leicht die neuen Indizes berechnen, indem wir die ungeraden Pufferstellen bis zur Überschreitung der alten Indexzahl entlangwandern. Der neue Wert ergibt sich dann sehr leicht durch Differenzbildung mit dem vorstehenden geraden ARRAY-Index.

C:index_packen und data_packen

In den eigentlichen Packroutinen sind nur noch, unter Auswechslung sämtlicher alter Indizes durch die neuen, die gültigen Datensätze in eine temporäre Datei zu kopieren. Hiernach erfolgt die Umbenennung der temporären in die alte Dateibezeichnung.

Vorausschau

So, das wär's !!! Die Bäume sind tot. In der nächsten Folge von Algorithmen & Datenstrukturen werden Sie Hashing-Verfahren kennenlernen, die mitunter auch recht hilfreich bei der DV sind. Sie können aufatmen. Die Listings werden dabei nämlich merklich kürzer und durchsichtiger. Bis dann!

(Sig)

```
 1: { Listing 6a)
 2:
 3:   Typdefinitionen fuer die Indexdatei und die
        Datenbank.
 4:
 5:   programmiert : Oktober 87
 6:   von          : Dirk Brockhaus
 7:   mit          : PASCAL+
 8: }
 9:
10:       key_type  = integer;
11:
12:       data_type = integer;
13:
14:       page_ptr  = integer;
15:
16:       data_ptr  = integer;
17:
18:       base_type = RECORD
19:                     flag : boolean;
20:                     next : data_ptr;
21:                     key  : key_type;
22:                     data : data_type;
23:                   END;
24:
25:       item_type = RECORD
26:                     key       : key_type;
27:                     record_nr : data_ptr;
28:                     ptr       : page_ptr;
29:                   END;
30:
31:       page_type = RECORD
32:                     flag : boolean;
33:                     anz  : integer;
34:                     ptr0 : page_ptr;
35:                     item : ARRAY [1..nn] OF item_type;
36:                   END;
```

```
 1: { Listing 6b)
 2:
 3:   Einige Routinen zur Unterhaltung der Datenbank.
 4:
 5:   programmiert : Oktober 87
 6:   von          : Dirk Brockhaus
 7:   mit          : PASCAL+
 8:
 9:   Funktionalitaeten :
10:
11:   ( dateilaenge : string -> long_integer x boolean )
12:     create_base : () -> FILE OF page_type x
          FILE OF base_type
13:     start_base  : () -> FILE OF page_type x
          FILE OF base_type x
14:                    page_ptr x page_ptr x data_ptr
15:     insert_data : FILE OF page_type x
           FILE OF base_type x key_type x
16:                    data_type x page_ptr ->
                          FILE OF page_type x
17:                    FILE OF base_type x page_ptr
18:     delete_data : FILE OF page_type x FILE OF base_type
           x key_type x
19:                    page_ptr -> FILE OF page_type x
                          FILE OF base_type x
20:                    boolean x page_ptr
21:   ( search_ptr  : FILE OF page_type x key_type x
               page_ptr -> data_ptr )
22:     search_first: FILE OF base_type x
             FILE OF page_type x key_type x
23:                    page_ptr -> data_ptr x data_type x
                          boolean
24:     search_next : FILE OF base_type x
            data_ptr -> data_type x
25:                    data_ptr x boolean
26:     print_tree  : FILE OF page_type x
             FILE OF base_type -> text
27: }
28:
29: FUNCTION dateilaenge(    name   : string;
30:                      VAR laenge : long_integer) :
                              boolean;
31:
32:   TYPE dta_type      = RECORD
33:               reserviert : PACKED ARRAY [0..21] OF byte;
34:               zeit       : integer;
35:               datum      : integer;
36:               groesse    : long_integer;
37:               name       : PACKED ARRAY [1..14] OF char;
38:                          END;
39:
40:        dta_ptr_type   = ^dta_type;
41:
42:        sys_str        = PACKED ARRAY [1..80] OF char;
43:
44:   VAR  dta_ptr  : dta_ptr_type;
45:        pfad     : sys_str;
46:
47:   FUNCTION fgetdta : dta_ptr_type;
48:     GEMDOS($2f);
49:
50:   FUNCTION fsfirst (VAR path : sys_str;
51:                         attr : integer) : integer;
52:     GEMDOS($4e);
53:
54:   PROCEDURE copy_sys(    str  : string;
55:                      VAR path : sys_str);
56:
57:     VAR i : integer;
58:
59:     BEGIN {copy_sys}
60:       FOR i:=1 TO length(str) DO
61:         path[i]:=str[i];
62:       FOR i:=length(str)+1 TO 80 DO
63:         path[i]:=chr(0);
64:     END;  {copy_sys}
65:
66:   BEGIN {dateilaenge}
67:     {$p-}
68:     dta_ptr:=fgetdta;
69:     copy_sys(name,pfad);
70:     dateilaenge:=fsfirst(pfad,$17)>=0;
71:     laenge:=dta_ptr^.groesse;
72:     {$p+}
73:   END;  {dateilaenge}
74:
75: PROCEDURE create_database;
76:
77:   VAR i : integer;
78:
79:   BEGIN {create_database}
80:     rewrite(index,index_name);
81:     close(index);
82:     rewrite(datenbank,database_name);
```

geht weiter...

Jetzt bringt der ATARI ST Ihr Büro in Form

ST-Kontor
Kaufmännische Anwendungen für alle ATARI ST-Computer
ST-Kontor
TOS-Manager
Kundenverwaltung
Lager + Fakturierung
Finanzbuchhaltung
Lohn + Gehalt
SYBEX
Finanzbuchhaltung
Software + Handbuch

Wußten Sie eigentlich, wieviel Arbeit Ihnen Ihr ST bei den täglichen Büro-Aufgaben abnehmen kann? Immer vorausgesetzt, daß Sie über eine Software verfügen, die alle nötigen Funktionen besitzt, komfortabel und schnell ist; wie die von der GFE R. Becker KG entwickelten Programme der Reihe SYBEX ST-Kontor (jeweils mit Trainingsbuch). Ihr ATARI ST (260–1040, MegaST 2–4) sollte einen Speicher von 1 MB RAM (für TOS-Manager und Kundenverwaltung reichen auch 512 KB) und das Betriebssystem TOS im ROM haben. Die mächtige FiBu benötigt ein double-sided Laufwerk mit 1 MB, optimal wäre eine Festplatte.

Natürlich finden Sie bei SYBEX noch mehr Software und Bücher zu Ihrem Rechner. Fordern Sie einfach unseren ATARI ST-Prospekt oder unser Gesamtverzeichnis an.

So hilft Ihnen der ST-Kontor TOS-Manager:

r ist die neue Verwaltungszentrale Ihres ATARI-Rechners. Von hier aus können Sie alle übrigen ST-Kontor-Programme ohne anges Suchen aufrufen. Viele Accessories erleichtern Ihnen die Arbeit: RAMDisk, Taschenrechner, Terminkalender, ASCII-abelle, Notizblock usw. Die Diskettenverwaltung ersetzt die GEM-Benutzeroberfläche und bietet ihnen viele zusätzliche Mög-chkeiten, wie: Formatieren und Kopieren von Disketten, Bearbeiten von Ordnern und Dateien, Suchen von Dateien, estaurieren gelöschter Dateien, Druckereinstellung. Ein Programm, mit dem Sie Ihre tägliche Arbeit einfacher und effizienter estalten.

est.-Nr. **3428, DM 98,–*/sFr. 98,–/S 872,–**

Und das kann die ST-Kontor Kundenverwaltung:

Als Basismodul der ST-Kontor-Reihe ist die Kundenverwaltung für die Verwaltung Ihrer Adressen-Bestände zuständig; nur Ihre Festplatte bzw. das RAM können dabei Grenzen setzen. Die wichtigsten Features im Überblick:

- Superschnelles Suchen (Suchen nach Von-Bis-Werten in max. 0,5 Sekunden)
- Optimierte Datenspeicherung
- Mehrere Datenfelder pro Adresse (Stammdaten, Zusatzdaten, Notizen u. a.)
- Umfangreiche Selektionsmöglichkeiten nach beliebig vielen Kriterien
- Ausgabe auf Datei/Monitor/Drucker als Liste, auf Etikett, als Serienbrief

Best.-Nr. **3429, DM 149,–*/sFr. 149,–/S 1326,–**

Ein Bündel arbeitssparender Features enthält die ST-Kontor Lagerverwaltung & Fakturierung:

Mit diesem starken Kombi-Programm bringen Sie Ihr Sortiment auf Vordermann: Stücklisten, Etiketten, Preislisten, Kataloge und umfangreiche Angebote mit Textbausteinen sind jetzt kein Problem mehr für Sie. Ebenso selbstverständlich sind verschiedene Lagerbestands-Bewertungsmethoden, Übersicht über den aktuellen Lagerbestand mit Bestellvorschlägen, Fakturierungsübersicht mit Offenen Posten, Teilen und Zusammenlegen von Fakturierungen – und die Übernahme in die Finanzbuchhaltung. Nicht zu vergessen die Bildschirmkasse mit dem aktuellen Kassenstand, Anzeige des Wechselgeldes und Ausdruck eines Kaufbelegs. Und betriebswirtschaftliche Funktionen zur Betriebsoptimierung werden direkt mitgeliefert.

Best.-Nr. **3430, DM 398,–*/sFr. 398,–/S 3542,–**



Den aktuellen EG-Richtlinien entspricht die ST-Kontor Finanzbuchhaltung:

Die mehrfirmen- und mandantenfähige FiBu verdaut große Buchungsmengen mit einer erstaunlichen Geschwindigkeit. Unter anderem können Sie von ihr erwarten:

- Kontenrahmen SKR 04 mit 6000 Konten + Einzelhandels-Kontenrahmen SKR 02
- Verschiedene Bilanzierungs- und Abschlußmöglichkeiten mit einfacher Kostenstellen-Rechnung, Anlagenspiegel, Abschreibungs-Verzeichnis und betriebswirtschaftlicher Auswertung
- Verbuchen von Sammelbelegen mit maximal 10 Soll- und 10 Habenkonten sowie 4 Mehrwertsteuer-Sätzen pro Buchung
- Abwicklung des Zahlungsverkehrs mit Offenen Posten und Mahnwesen
- Übernahmedatei für Fakturierung/Lohn & Gehalt mit Korrekturmöglichkeiten

Best.-Nr. **3431, DM 498,–*/sFr. 498,–/S 4432,–**

Das erledigt ST-Kontor Lohn & Gehalt für Sie:

In Vorbereitung

- Verwaltung der Mitarbeiterdaten samt Ausfüllen aller Versicherungs- und Finanzamtsformulare
- Alle Lohn- und Gehaltskonten auf einen Blick
- Verwaltung von Zeitkonten einschließlich Krankheits- und Urlaubstagen
- Abrechnung von Vermögensbildung, Direktversicherung und Lohnpfändung
- Komplette Erstellung der Lohnsteuer-Jahresdaten ohne Rechenaufwand
- Abwickeln des Zahlungsverkehrs mit Mitarbeitern, Ämtern und Versicherungen inklusive Berechnung der Zahlungsbeträge und Vorbereitung der Buchung. Die Daten können von der ST-Kontor FiBu übernommen und weiter bearbeitet werden.

Best.-Nr. **3432, DM 198,–*/sFr. 198,–/S 1760,–**

* Unverbindliche Preisempfehlung

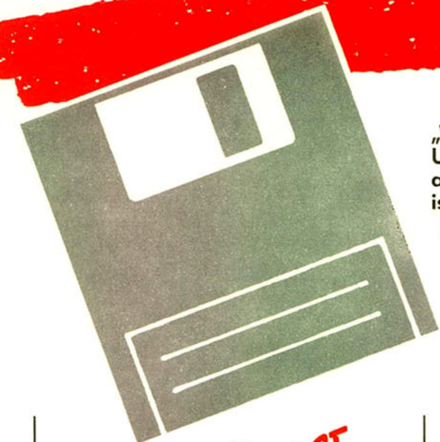
Übrigens: SYBEX sucht ständig gute Buch- und Software-Autoren. Interessiert? Dann kontakten Sie bitte Ralf Lieder, Tel. 0211/618 02 20.



```
 83:      close(datenbank);
 84:     END;  {create_database}
 85:
 86:   PROCEDURE start_database;
 87:
 88:     VAR laenge : long_integer;
 89:
 90:     BEGIN {start_database}
 91:       IF dateilaenge(index_name,laenge) THEN
 92:         last_index:=int(laenge) DIV sizeof(page_type)-1;
 93:       IF dateilaenge(database_name,laenge) THEN
 94:         last_data:=int(laenge) DIV sizeof(base_type)-1;
 95:       reset(index,index_name);
 96:       reset(datenbank,database_name);
 97:       IF last_index=leer THEN
 98:         tree:=leer
 99:       ELSE
100:         tree:=wurzel;
101:     END;  {start_database}
102:
103:   PROCEDURE insert_data(new_key   : key_type;
104:                         new_data : data_type);
105:
106:     VAR h    : boolean;
107:         u    : item_type;
108:
109:     PROCEDURE search(     x : key_type;
110:                           a : page_ptr;
111:                       VAR h : boolean;
112:                       VAR v : item_type);
113:
114:       VAR k ,
115:           l ,
116:           r : integer;
117:           u : item_type;
118:
119:       PROCEDURE insert;
120:
121:         VAR i       : integer;
122:             b       : page_type;
123:
124:         BEGIN {insert}
125:           get(index,a);
126:           WITH index^ DO
127:             IF anz<nn THEN
128:               BEGIN
129:                 anz:=anz+1;
130:                 h:=false;
131:                 FOR i:=anz DOWNTO r+2 DO
132:                   item[i]:=item[i-1];
133:                 item[r+1]:=u;
134:                 put(index,a);
135:               END
136:             ELSE
137:               BEGIN
138:                 IF r<=n THEN
139:                   BEGIN
140:                     IF r=n THEN
141:                       v:=u
142:                     ELSE
143:                       BEGIN
144:                         v:=item[n];
145:                         FOR i:=n DOWNTO r+2 DO
146:                           item[i]:=item[i-1];
147:                         item[r+1]:=u;
148:                       END;
149:                     FOR i:=1 TO n DO
150:                       b.item[i]:=item[i+n];
151:                   END
152:                 ELSE
153:                   BEGIN
154:                     r:=r-n;
155:                     v:=item[n+1];
156:                     FOR i:=1 TO r-1 DO
157:                       b.item[i]:=item[i+n+1];
158:                     b.item[r]:=u;
159:                     FOR i:=r+1 TO n DO
160:                       b.item[i]:=item[i+n];
161:                   END;
162:                 anz:=n;
163:                 put(index,a);
164:                 b.anz:=n;
165:                 b.ptr0:=v.ptr;
166:               index^:=b;
167:               index^.flag:=true;
168:               last_index:=last_index+1;
169:               put(index,last_index);
170:               v.ptr:=last_index;
171:             END;
172:         END;  {insert}
173:
174:       BEGIN {search}
175:         IF a=leer THEN
176:           BEGIN
177:             h:=true;
178:             WITH v DO
179:               BEGIN
180:                 key:=x;
181:                 ptr:=leer;
182:                 record_nr:=last_data;
183:               END;
184:           END
185:         ELSE
186:           BEGIN
187:             get(index,a);
188:             WITH index^ DO
189:               BEGIN
190:                 l:=1;
191:                 r:=anz;
192:                 REPEAT
193:                   k:=(l+r) DIV 2;
194:                   IF x<=item[k].key THEN
195:                     r:=k-1;
196:                   IF x>=item[k].key THEN
197:                     l:=k+1;
198:                 UNTIL r<l;
199:                 IF l-r>1 THEN
200:                   BEGIN
201:                     get(datenbank,last_data);
202:                     datenbank^.next:=item[k].record_nr;
203:                     put(datenbank,last_data);
204:                     item[k].record_nr:=last_data;
205:                     put(index,a);
206:                     h:=false;
207:                   END
208:                 ELSE
209:                   BEGIN
210:                     IF r=0 THEN
211:                       search(x,ptr0,h,u)
212:                     ELSE
213:                       search(x,item[r].ptr,h,u);
214:                     IF h THEN
215:                       insert;
216:                   END;
217:               END;
218:           END;
219:       END;  {search}
220:
221:     BEGIN {insert_data}
222:       WITH datenbank^ DO
223:         BEGIN
224:           flag:=true;
225:           next:=leer;
226:           key:=new_key;
227:           data:=new_data;
228:         END;
229:       last_data:=last_data+1;
230:       put(datenbank,last_data);
231:       search(new_key,tree,h,u);
232:       IF h THEN
233:         BEGIN
234:           IF last_index<>leer THEN
235:             get(index,0);
236:           index^.flag:=true;
237:           last_index:=last_index+1;
238:           put(index,last_index);
239:           tree:=wurzel;
240:           WITH index^ DO
241:             BEGIN
242:               anz:=1;
243:               IF last_index=0 THEN
244:                 ptr0:=-1
245:               ELSE
246:                 ptr0:=last_index;
247:               item[1]:=u;
248:             END;
249:           put(index,wurzel);
250:         END;
251:     END;  {insert_data}
252:
253:   FUNCTION delete_data(key : key_type) : boolean;
254:
```

geht weiter....

PREISFRAGE!



„Wir machen Spitzentechnologie preiswert", lobt Atari seine ST-Reihe. Und das zu recht. Doch warum soll, was für die Hardware gilt, nicht auch bei der Software möglich sein? Schließlich gilt auch hier: Qualität ist keine Frage des Preises und der Preis eher eine Frage der Stückzahl. Deswegen vertreiben wir ausgereifte Programme in hohen Stückzahlen. Zu Preisen, die Ihnen den heißersehnten Einstieg leicht machen.

BECKERbase ST

Haben Sie ihn nicht schon länger gesucht – den Einstieg in ein professionelles Datenbank-Programm? Hier ist er. BECKERbase ST. Völlig unproblematische Installation und Bedienung. Zwei integrierte Programmiersprachen sorgen für eine hohe Flexibilität in der Anpassung an spezifische Benutzerbedürfnisse. Besonderer Vorteil: die Datei-Beziehungen sind als Netzwerk angelegt.

BECKERbase ST in Kürze:
Unbegrenzte Anzahl von Dateien pro Datenbank – maximal 65535 Sätze pro Datei – Satzlänge max. 64 KByte – unbegrenzte Anzahl Felder pro Datensatz – maximal 255 Zeichen pro Feld – integrierte, leicht erlernbare Datenbanksprache DDL (Data Definition Language), TDL (Transaction Definition Language) zur individuellen Datenbankerstellung – separater, voll in GEM integrierter Maskeneditor – implementierte Import-Funktion zur einfachen Übernahme von Datenbeständen anderer Programme – Export-Funktion zur Übergabe von Daten z. B. an eine Textverarbeitung – Kompatibilität zu BECKERbase PC, so daß nicht nur Daten ausgetauscht werden können, sondern sogar die Literatur über BECKERbase PC verwendet werden kann. Empfohlen wird ROM-TOS und 1MByte RAM.

BECKERbase ST DM 99,–

BECKERtools ST

Eine Sammlung kleiner, hilfreicher GEM-Programme als Accessories, die immer da sind, wenn der Anwender sie braucht. Dabei sind alle Tools jederzeit nachladbar, selbst wenn Sie sich innerhalb eines Programms befinden.

BECKERtools ST in Kürze:
Hebt praktisch alle Beschränkungen für die Accessory-Anwendung auf – belegt nur einen Eintrag im DESK-Menü, beinhaltet aber die Möglichkeit, bis zu 30 Tools (Accessories) aufzurufen – diese Tools können zu beliebiger Zeit nachgeladen werden, nicht nur beim Booten – Tools können jederzeit wieder entfernt werden – folgende Tools gehören unter anderem zum Programmpaket: DIR, übersichtliche Anzeige des Inhaltsverzeichnisses mit Editierung der Dateiparameter (Name, Datum, Zeit) – EDDY, praktischer Editor für ASCII-Dateien – FILECOPY kopiert Dateien – FORMAT, formatiert bis zu max. 820 KByte/Diskette – ORDNER, erzeugt Dateien – UNDELETE, restauriert gelöschte Dateien, falls irgend möglich – WIPEFILE, löscht eine Datei unwiderruflich – Harddisk-Backup. Lauffähig mit Monochrom-Monitor.

BECKERtools ST DM 99,–

PLATINE ST

Ist das wirklich das Profi-Programm, das Sie schon immer haben wollten, aber sich bisher nicht leisten konnten? Ja, Sie träumen nicht. Mit PLATINE ST erstellen Sie elektronische Leiterplatten mit einem Minimum an Kosten, Zeit und Arbeitsaufwand.

PLATINE ST in Kürze:
Pro Schaltung bis zu 250 Bauteile und 1100 Verbindungen – unterstützt Leiterbahnen mit 45- und 90-Grad-Segmenten und zwei verschiedene Leiterbahnstärken – ein oder zwei Layer, bis Europakartenformat – selbstdefinierte Bauteile können verwendet werden – der Auto-Router kann durch umfangreiche Optionen beeinflußt werden – flexible und interaktive Bauteileplazierung – volle Menüsteuerung – Platinenüberprüfung, Verbindungslisten mit bestehenden Verbindungen – Bauteile können in 90-Grad-Schritten gedreht werden – übersichtliche Bearbeitung der Platine durch übereinanderliegende Layer und Bestückungsplan – Ausdruck ist sehr genau und kann zur fototechnischen Bearbeitung weiterverwendet werden – Treiber u. a. für die Drucker NEC P6/P7, Toshiba P 351 und HP-GL Plotter – mit ausführlichem deutschen Handbuch – lauffähig mit Monochrom-Monitor.

PLATINE ST DM 99,–

DATA BECKER
Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010

BESTELL-COUPON
Einsenden an: DATA BECKER · Merowingerstr. 30 · 4000 Düsseldorf 1
Bitte senden Sie mir:
zzgl. DM 5,– Versandkosten unabhängig von der bestellten Stückzahl
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei
Name
Straße
Ort

```
255:   VAR del_list : data_ptr;
256:       q        : page_ptr;
257:       h        : boolean;
258:
259:    PROCEDURE search(     x : key_type;
260:                          a : page_ptr;
261:                      VAR h : boolean);
262:
263:      VAR q : page_ptr;
264:          i ,
265:          k ,
266:          l ,
267:          r : integer;
268:
269:      PROCEDURE underflow(     c ,
270:                               a : page_ptr;
271:                               s : integer;
272:                           VAR h : boolean);
273:
274:        VAR b  : page_ptr;
275:            i  ,
276:            k  ,
277:            mb ,
278:            mc : integer;
279:            ic ,
280:            ia : page_type;
281:
282:        BEGIN {underflow}
283:          get(index,c);
284:          ic:=index^;
285:          get(index,a);
286:          ia:=index^;
287:          mc:=ic.anz;
288:          IF s<mc THEN
289:            BEGIN
290:              s:=s+1;
291:              b:=ic.item[s].ptr;
292:              get(index,b);
293:              mb:=index^.anz;
294:              k:=(mb-n+1) DIV 2;
295:              ia.item[n]:=ic.item[s];
296:              ia.item[n].ptr:=index^.ptr0;
297:              IF k>0 THEN
298:                BEGIN
299:                  FOR i:=1 TO k-1 DO
300:                    ia.item[i+n]:=index^.item[i];
301:                  ic.item[s]:=index^.item[k];
302:                  ic.item[s].ptr:=b;
303:                  index^.ptr0:=index^.item[k].ptr;
304:                  mb:=mb-k;
305:                  FOR i:=1 TO mb DO
306:                    index^.item[i]:=index^.item[i+k];
307:                  index^.anz:=mb;
308:                  ia.anz:=n-1+k;
309:                  h:=false;
310:                END
311:              ELSE
312:                BEGIN
313:                  FOR i:=1 TO n DO
314:                    ia.item[i+n]:=index^.item[i];
315:                  FOR i:=s TO mc-1 DO
316:                    ic.item[i]:=ic.item[i+1];
317:                  ia.anz:=nn;
318:                  ic.anz:=mc-1;
319:                  h:=mc<=n;
320:                  index^.flag:=false;
321:                END
322:            END
323:          ELSE
324:            BEGIN
325:              IF s=1 THEN
326:                b:=ic.ptr0
327:              ELSE
328:                b:=ic.item[s-1].ptr;
329:              get(index,b);
330:              mb:=index^.anz+1;
331:              k:=(mb-n) DIV 2;
332:              IF k>0 THEN
333:                BEGIN
334:                  FOR i:=n-1 DOWNTO 1 DO
335:                    ia.item[i+k]:=ia.item[i];
336:                  ia.item[k]:=ic.item[s];
337:                  ia.item[k].ptr:=ia.ptr0;
338:                  mb:=mb-k;
339:                  FOR i:=k-1 DOWNTO 1 DO
340:                    ia.item[i]:=index^.item[i+mb];
341:                  ia.ptr0:=index^.item[mb].ptr;
342:                  ic.item[s]:=index^.item[mb];
343:                  ic.item[s].ptr:=a;
344:                  index^.anz:=mb-1;
345:                  ia.anz:=n-1+k;
346:                  h:=false;
347:                END
348:              ELSE
349:                BEGIN
350:                  index^.item[mb]:=ic.item[s];
351:                  index^.item[mb].ptr:=ia.ptr0;
352:                  FOR i:=1 TO n-1 DO
353:                    index^.item[i+mb]:=ia.item[i];
354:                  index^.anz:=nn;
355:                  ic.anz:=mc-1;
356:                  h:=mc<=n;
357:                  ia.flag:=false;
358:                END;
359:            END;
360:          put(index,b);
361:          index^:=ia;
362:          put(index,a);
363:          index^:=ic;
364:          put(index,c);
365:        END;  {underflow}
366:
367:      PROCEDURE del(     p : page_ptr;
368:                     VAR h : boolean);
369:
370:        VAR q  : page_ptr;
371:            ip : page_type;
372:
373:        BEGIN {del}
374:          get(index,p);
375:          ip:=index^;
376:          WITH ip DO
377:            BEGIN
378:              q:=item[anz].ptr;
379:              IF q<>leer THEN
380:                BEGIN
381:                  del(q,h);
382:                  IF h THEN
383:                    underflow(p,q,anz,h);
384:                END
385:              ELSE
386:                BEGIN
387:                  get(index,a);
388:                  item[anz].ptr:=index^.item[k].ptr;
389:                  index^.item[k]:=item[anz];
390:                  anz:=anz-1;
391:                  h:=anz<n;
392:                  put(index,a);
393:                  index^:=ip;
394:                  put(index,p);
395:                END;
396:            END;
397:        END;  {del}
398:
399:      BEGIN {search}
400:        IF a=leer THEN
401:          BEGIN
402:            h:=false;
403:            del_list:=leer;
404:            delete_data:=false;
405:          END
406:        ELSE
407:          BEGIN
408:            get(index,a);
409:            WITH index^ DO
410:              BEGIN
411:                l:=1;
412:                r:=anz;
413:                REPEAT
414:                  k:=(l+r) DIV 2;
415:                  IF x<=item[k].key THEN
416:                    r:=k-1;
417:                  IF x>=item[k].key THEN
418:                    l:=k+1;
419:                UNTIL r<l;
420:                IF r=0 THEN
421:                  q:=ptr0
422:                ELSE
423:                  q:=item[r].ptr;
424:                IF l-r>1 THEN
425:                BEGIN
426:                  del_list:=item[k].record_nr;
427:                  IF q=leer THEN
```

geht weiter...

```
428:                  BEGIN
429:                    anz:=anz-1;
430:                    h:=anz<n;
431:                    FOR i:=k TO anz DO
432:                      item[i]:=item[i+1];
433:                    put(index,a);
434:                  END
435:                ELSE
436:                  BEGIN
437:                    del(q,h);
438:                    IF h THEN
439:                      underflow(a,q,r,h);
440:                  END;
441:              END
442:            ELSE
443:              BEGIN
444:                search(x,q,h);
445:                IF h THEN
446:                  underflow(a,q,r,h);
447:              END;
448:          END;
449:      END;
450:    END;  {search}
451:
452:  BEGIN {delete_data}
453:    IF tree<>leer THEN
454:      BEGIN
455:        delete_data:=true;
456:        search(key,tree,h);
457:        IF h THEN
458:          BEGIN
459:            get(index,tree);
460:            IF index^.anz=0 THEN
461:              IF index^.ptr0=leer THEN
462:                BEGIN
463:                  tree:=leer;
464:                  index^.flag:=false;
465:                  put(index,wurzel);
466:                END
467:              ELSE
468:                BEGIN
469:                  q:=index^.ptr0;
470:                  get(index,q);
471:                  put(index,tree);
472:                  index^.flag:=false;
473:                  put(index,q);
474:                END;
475:          END;
476:        WHILE del_list<>leer DO
477:          BEGIN
478:            get(datenbank,del_list);
479:            datenbank^.flag:=false;
480:            put(datenbank,del_list);
481:            del_list:=datenbank^.next;
482:          END;
483:      END;
484:  END;  {delete_data}
485:
486: FUNCTION search_ptr(a    : page_ptr;
487:                     key : key_type) : data_ptr;
488:
489:   VAR q : page_ptr;
490:       l ,
491:       r ,
492:       k : integer;
493:
494:   BEGIN {search_ptr}
495:     IF a=leer THEN
496:       search_ptr:=leer
497:     ELSE
498:       BEGIN
499:         get(index,a);
500:         WITH index^ DO
501:           BEGIN
502:             l:=1;
503:             r:=anz;
504:             REPEAT
505:               k:=(l+r) DIV 2;
506:               IF key<=item[k].key THEN
507:                 r:=k-1;
508:               IF key>=item[k].key THEN
509:                 l:=k+1;
510:             UNTIL r<l;
511:             IF r=0 THEN
512:               q:=ptr0
513:             ELSE
514:               q:=item[r].ptr;
515:             IF l-r>1 THEN
516:               search_ptr:=item[k].record_nr
517:             ELSE
518:               search_ptr:=search_ptr(q,key);
519:           END;
520:       END;
521:   END;  {search_ptr}
522:
523: FUNCTION search_first(     key   : key_type;
524:                        VAR data : data_type) : boolean;
525:
526:   BEGIN {search_first}
527:     next_list:=search_ptr(tree,key);
528:     search_first:=next_list<>leer;
529:     IF next_list<>leer THEN
530:       BEGIN
531:         get(datenbank,next_list);
532:         data:=datenbank^.data;
533:         next_list:=datenbank^.next;
534:       END;
535:   END;  {search_first}
536:
537: FUNCTION search_next(VAR data : data_type) : boolean;
538:
539:   BEGIN {search_next}
540:     search_next:=next_list<>leer;
541:     IF next_list<>leer THEN
542:       BEGIN
543:         get(datenbank,next_list);
544:         data:=datenbank^.data;
545:         next_list:=datenbank^.next;
546:       END;
547:   END;  {search_next}
548:
549: PROCEDURE print_tree(tree   : page_ptr;
550:                      tiefe : integer);
551:
552:   VAR i     : integer;
553:       copy : page_type;
554:
555:   BEGIN {print_tree}
556:     IF tree<>leer THEN
557:       BEGIN
558:         write(tiefe,': ');
559:         get(index,tree);
560:         WITH index^ DO
561:           BEGIN
562:             FOR i:=1 TO anz DO
563:               BEGIN
564:                 write(item[i].key,' (');
565:                 get(datenbank,item[i].record_nr);
566:                 write(datenbank^.data);
567:                 WHILE datenbank^.next<>leer DO
568:                   BEGIN
569:                     get(datenbank,datenbank^.next);
570:                     write('/',datenbank^.data);
571:                   END;
572:                 write(') ');
573:               END;
574:             writeln;
575:             copy:=index^;
576:             print_tree(ptr0,tiefe+1);
577:             FOR i:=1 TO copy.anz DO
578:               print_tree(copy.item[i].ptr,tiefe+1);
579:           END;
580:       END;
581:   END;  {print_tree}
```

Listing 6b (Ende)

```
1:  { Listing 6c)
2:
3:    Eine Testumgebung fuer unsere Datenbank mit Bayerbaum
        indizierung.
4:
5:    programmiert : Oktober 87
6:    von          : Dirk Brockhaus
7:    mit          : PASCAL+
8:  }
9:
10:  PROGRAM test_b_baeume (input,output);
11:
12: CONST nn             = 4;
13:       n              = 2;
```

geht weiter....

Festplattensysteme für ATARI ST und MEGA ST

30 MB Kapazität
DM 1498,–*

40 MB Kapazität
DM 2198,–*

60 MB Kapazität
DM 2398,–*

...UND PLÖTZLICH IST DIE ENT-SCHEIDUNG GANZ EINFACH: VER-GLEICHEN SIE PREIS UND LEISTUNG

* unverbindliche Preisempfehlung

vortex HDplus Festplatten-Systeme von 20 bis 120 MB!

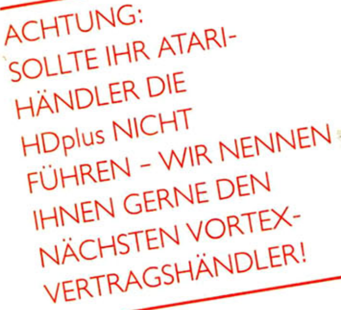
Das hat es bis jetzt noch nicht gegeben: Ein Festplatten-Programm für den ATARI ST bzw. MEGA ST mit Kapazitäten von 20 bis 120 MB formatierte Speicherkapazität!

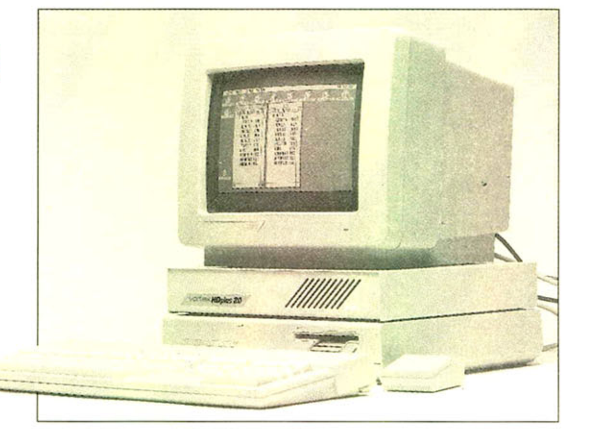
Weitere Vorteile

- Cache-Memory
- Auto-Parker
- bis zu 16 Partitionen
- Disketten-Backup-Programm
- bootfähig

Natürlich im ATARI-Design und in vortex-Qualität. Komplett anschlußfertig mit System-Diskette, Buskabel und deutschem Handbuch.

Holen Sie sich die kompletten Informationen. Sofort!







...UND PLÖTZLICH LEISTET IHR COMPUTER MEHR

ST- 4/88

I·N·F·O-S·C·H·E·C·K

Senden Sie mir umgehend alle Informationen über Ihr HDplus-Programm und nennen Sie mir den nächstgelegenen vortex-Vertragshändler.

vortex Computersysteme GmbH
Falterstraße 51–53 · 7101 Flein · Telefon (07131) 52061

```
14:         leer           = -1;
15:         wurzel         = 0;
16:          database_name = 'TEST.DBF';
17:          index_name    = 'TEST.NDX';
18:
19:   TYPE  {$i bayertyp.pas}
20:
21:  VAR    index      : FILE OF page_type;
22:         datenbank  : FILE OF base_type;
23:         last_data  ,
24:         next_list  : data_ptr;
25:         last_index ,
26:         tree       : page_ptr;
27:
28:         {$i bayersub.pas}
29:
30:   FUNCTION random : integer;
31:
32:     FUNCTION rnd : integer;
33:       XBIOS(17);
34:
35:     BEGIN {random}
36:       random:=abs(rnd) MOD 100;
37:     END;  {random}
38:
39:   PROCEDURE do_main;
40:
41:     VAR ch1  ,
42:         ch2  : char;
43:         key  : key_type;
44:         data : data_type;
45:         i    ,
46:         anz  : integer;
47:
48:     BEGIN {do_main}
49:       REPEAT
50:         writeln;
51:          writeln('Testumgebung fuer Bayer-Baeume:');
52:         writeln;
53:          writeln('(1) Einfuegen');
54:          writeln('(2) Loeschen');
55:          writeln('(3) Suchen');
56:          writeln('(4) Baumausgabe');
57:          writeln('(5) Loeschen aller Daten');
58:          writeln('(6) Random-Daten einfuegen');
59:         writeln;
60:          writeln('(0) Beenden');
61:         writeln;
62:         write('Ihre Wahl -> '); read(ch1);
63:         writeln;
64:         writeln;
65:         CASE ch1 OF
66:           '1' : BEGIN
67:                   writeln;
68:                   writeln;
69:                    writeln('Einfuegen:');
70:                   writeln;
71:                    write('Schluessel (integer) ? ');
                           readln(key);
72:                   write('Daten       (integer) ? ');
                           readln(data);
73:                    insert_data(key,data);
74:                    writeln('Daten eingefuegt !');
75:                   readln;
76:                 END;
77:           '2' : BEGIN
78:                   writeln;
79:                   writeln;
80:                    writeln('Loeschen:');
81:                   writeln;
82:                    write('Schluessel (integer) ? ');
                           readln(key);
83:                    IF NOT delete_data(key) THEN
84:                       writeln('Dieser Schluessel existiert
                             nicht !')
85:                   ELSE
86:                       writeln('Eintrag geloescht !');
87:                   readln;
88:                 END;
89:           '3' : BEGIN
90:                   writeln;
91:                   writeln;
92:                    writeln('Suche:');
93:                   writeln;
94:                    write('Schluessel (integer) ? ');
                           readln(key);
95:                    IF search_first(key,data) THEN
96:                     BEGIN
97:                       write('(',data);
98:                       WHILE search_next(data) DO
99:                         write('\',data);
100:                      writeln(')');
101:                       writeln('Ausgabe komplett !');
102:                    END
103:                 ELSE
104:                    writeln('Eintrag nicht gefunden !');
105:                 readln;
106:               END;
107:          '4' : BEGIN
108:                  writeln;
109:                  writeln;
110:                   writeln('Ausgabe des gesamten
                              Baumes:');
111:                  writeln;
112:                   write('Ausgabe auf (B)ildschirm oder
                              (D)rucker');
113:                  read(ch2);
114:                  IF ch2 IN ['d','D'] THEN
115:                     rewrite(output,'PRN:');
116:                  writeln;
117:                  writeln;
118:                   print_tree(tree,1);
119:                  writeln;
120:                  writeln;
121:                   rewrite(output,'CON:');
122:                   writeln('Ausgabe komplett !');
123:                  readln;
124:                END;
125:          '5' : BEGIN
126:                  create_database;
127:                  start_database;
128:                END;
129:          '6' : BEGIN
130:                  writeln;
131:                  writeln;
132:                   writeln('Zufallszahlen einfuegen:');
133:                  writeln;
134:                   write('Anzahl ? '); readln(anz);
135:                  FOR i:=1 TO anz DO
136:                    BEGIN
137:                      key:=random;
138:                      data:=random;
139:                       insert_data(key,data);
140:                    END;
141:                   writeln('Einfuegen abgeschlossen.');
142:                  readln;
143:                END;
144:        END;
145:      UNTIL ch1='0';
146:    END;  {do_main}
147:
148:  BEGIN
149:    create_database;
150:    start_database;
151:    do_main;
152:  END.
```

Listing 6c (Ende)

```
1:  { Listing 6d)
2:
3:     Packen einer Datenbank mit Bayerbaumindizierung.
4:
5:     programmiert : Oktober '87
6:     von          : Dirk Brockhaus
7:     mit          : PASCAL+ (CCD)
8:  }
9:
10:  PROGRAM pack (input,output);
11:
12: CONST nn             = 4;
13:       n              = 2;
14:       leer           = -1;
15:       wurzel         = 0;
16:        database_name = 'TEST.DBF';
17:        index_name    = 'TEST.NDX';
18:
19:  TYPE  {$i bayertyp.pas}
20:
21:  VAR   index      : FILE OF page_type;
22:        datenbank  : FILE OF base_type;
23:
24:  FUNCTION index_belegung : real;
25:
26:    VAR gesamt ,
```

geht weiter...

```
 27:        belegt : integer;
 28:
 29:    BEGIN {index_belegung}
 30:       reset(index,index_name);
 31:      gesamt:=0;
 32:      belegt:=0;
 33:      WHILE NOT eof(index) DO
 34:        BEGIN
 35:          gesamt:=gesamt+1;
 36:          IF index^.flag THEN
 37:            belegt:=belegt+1;
 38:          get(index);
 39:        END;
 40:      IF gesamt<>0 THEN
 41:         index_belegung:=belegt/gesamt
 42:      ELSE
 43:         index_belegung:=1;
 44:    END;  {index_belegung}
 45:
 46:  FUNCTION data_belegung : real;
 47:
 48:    VAR gesamt ,
 49:        belegt : integer;
 50:
 51:    BEGIN {data_belegung}
 52:       reset(datenbank,database_name);
 53:      gesamt:=0;
 54:      belegt:=0;
 55:      WHILE NOT eof(datenbank) DO
 56:        BEGIN
 57:          gesamt:=gesamt+1;
 58:          IF datenbank^.flag THEN
 59:            belegt:=belegt+1;
 60:          get(datenbank);
 61:        END;
 62:      IF gesamt<>0 THEN
 63:         data_belegung:=belegt/gesamt
 64:      ELSE
 65:         data_belegung:=1;
 66:    END;  {data_belegung}
 67:
 68:  PROCEDURE packen;
 69:
 70:    TYPE puffer = RECORD
 71:                    anz : integer;
 72:                    b   : ARRAY [1..1000] OF integer;
 73:                  END;
 74:
 75:    VAR  p_index ,
 76:         p_data  : puffer;
 77:
 78:    PROCEDURE abstand_besorgen;
 79:
 80:      VAR i    ,
 81:          anz : integer;
 82:          on  : boolean;
 83:
 84:      BEGIN {abstand_besorgen}
 85:        anz:=0;
 86:        i:=1;
 87:        on:=true;
 88:        p_data.anz:=0;
 89:        WHILE NOT eof(datenbank) DO
 90:          BEGIN
 91:            IF datenbank^.flag=on THEN
 92:              anz:=anz+1
 93:            ELSE
 94:              BEGIN
 95:                p_data.anz:=p_data.anz+1;
 96:                p_data.b[i]:=anz;
 97:                anz:=1;
 98:                on:=NOT on;
 99:                i:=i+1;
100:              END;
101:            get(datenbank);
102:          END;
103:        anz:=0;
104:        i:=1;
105:        on:=true;
106:        p_index.anz:=0;
107:        WHILE NOT eof(index) DO
108:          BEGIN
109:            IF index^.flag=on THEN
110:              anz:=anz+1
111:            ELSE
112:              BEGIN
113:                p_index.anz:=p_index.anz+1;
114:                p_index.b[i]:=anz;
115:                anz:=1;
116:                on:=NOT on;
117:                i:=i+1;
118:              END;
119:            get(index);
120:          END;
121:      END;  {abstand_besorgen}
122:
123:    PROCEDURE abstand_kumulieren(VAR p : puffer);
124:
125:      VAR i : integer;
126:
127:      BEGIN {abstand_kumulieren}
128:        WITH p DO
129:          BEGIN
130:            i:=3;
131:            WHILE i<=anz DO
132:              BEGIN
133:                b[i]:=b[i]+b[i-1]+b[i-2];
134:                i:=i+2;
135:              END;
136:            i:=4;
137:            WHILE i<=anz DO
138:              BEGIN
139:                b[i]:=b[i]+b[i-2];
140:                i:=i+2;
141:              END;
142:          END;
143:      END;  {abstand_kumulieren}
144:
145:    FUNCTION new_index(old : page_ptr) : page_ptr;
146:
147:      VAR i : integer;
148:
149:      BEGIN {new_index}
150:        IF old<>leer THEN
151:          WITH p_index DO
152:            BEGIN
153:              i:=1;
154:              WHILE (b[i]<old) AND (i<=anz) DO
155:                i:=i+2;
156:              IF i=1 THEN
157:                new_index:=old
158:              ELSE
159:                IF i-1<=anz THEN
160:                  new_index:=old-b[i-1]
161:                ELSE
162:                  IF odd(anz) THEN
163:                    new_index:=old-b[anz]
164:                  ELSE
165:                    new_index:=old-b[anz-1];
166:            END
167:        ELSE
168:          new_index:=leer;
169:      END;  {new_index}
170:
171:    FUNCTION new_data(old : data_ptr) : data_ptr;
172:
173:      VAR i : integer;
174:
175:      BEGIN {new_data}
176:        IF old<>leer THEN
177:          WITH p_data DO
178:            BEGIN
179:              i:=1;
180:              WHILE (b[i]<old) AND (i<=anz) DO
181:                i:=i+2;
182:              IF i=1 THEN
183:                new_data:=old
184:              ELSE
185:                IF i-1<=anz THEN
186:                  new_data:=old-b[i-1]
187:                ELSE
188:                  IF odd(anz) THEN
189:                    new_data:=old-b[anz]
190:                  ELSE
191:                    new_data:=old-b[anz-1];
192:            END
193:        ELSE
194:          new_data:=leer;
195:      END;  {new_data}
196:
197:    PROCEDURE index_packen;
198:
```

geht weiter....

```
199:      VAR t : FILE OF page_type;
200:          i : integer;
201:
202:      BEGIN {index_packen}
203:        reset(index,index_name);
204:        rewrite(t,'temp');
205:        WHILE NOT eof(index) DO
206:          BEGIN
207:            IF index^.flag THEN
208:              BEGIN
209:                t^:=index^;
210:                WITH t^ DO
211:                  BEGIN
212:                    ptr0:=new_index(ptr0);
213:                    FOR i:=1 TO anz DO
214:                      BEGIN
215:                        item[i].ptr:=
                              new_index(item[i].ptr);
216:                        item[i].record_nr:=
                              new_data(item[i].record_nr);
217:                      END;
218:                  END;
219:                put(t);
220:              END;
221:            get(index);
222:          END;
223:        rename(t,index);
224:      END; {index_packen}
225:
226:    PROCEDURE data_packen;
227:
228:      VAR t : FILE OF base_type;
229:          i : integer;
230:
231:      BEGIN {data_packen}
232:        reset(datenbank,database_name);
233:        rewrite(t,'temp');
234:        WHILE NOT eof(datenbank) DO
235:          BEGIN
236:            IF datenbank^.flag THEN
237:              BEGIN
238:                t^:=datenbank^;
239:                t^.next:=new_data(t^.next);
240:                put(t);
241:              END;
242:            get(datenbank);
243:          END;
244:        rename(t,datenbank);
245:      END; {data_packen}
246:
247:    BEGIN {packen}
248:      reset(datenbank,database_name);
249:      reset(index,index_name);
250:      abstand_besorgen;
251:      abstand_kumulieren(p_index);
252:      abstand_kumulieren(p_data);
253:      index_packen;
254:      data_packen;
255:    END; {packen}
256:
257:  BEGIN
258:    writeln('Packen der Dateien ',database_name,' und ',
          index_name);
259:    writeln;
260:    writeln('Belegungsgrad ',database_name,' = ',
            data_belegung*100:6:2,'%');
261:
262:    writeln('Belegungsgrad ',index_name,' = ',
            index_belegung*100:6:2,'%');
263:
264:    writeln;
265:    writeln('Beginne Packen.');
266:    packen;
267:    writeln('Packen beendet.');
268:    writeln;
269:    writeln('Teste Belegungsgrad:');
270:    writeln;
271:    writeln('Belegungsgrad ',database_name,' = ',
272:            data_belegung*100:6:2,'%');
273:    writeln('Belegungsgrad ',index_name,' = ',
274:            index_belegung*100:6:2,'%');
275:    writeln;
276:  END.
```

Listing 6d (Ende)

ENDE

DYNASTIEN

The British Way of Pascal im Vergleich mit einem Produkt aus deutschen Landen, frisch auf den Tisch.

Zwei der ältesten Pascal-Systeme für den ST, das Prospero ProPascal-System (das jetzt den Untertitel 'for GEM' trägt, um seine Anpassung an das DR GEM-System zu dokumentieren - übrigens auch für den MS-DOS Bereich erhältlich) und das CCD Pascal Plus liegen seit kurzem in neuen Versionen vor. Die Veteranen haben sich, diese Bemerkung sei vorab erlaubt, wirklich gut gehalten, was ihrem Überleben auch sehr förderlich sein dürfte, angesichts einer ganzen Flut von angekündigten neuen Pascal-Compilern und Entwicklungssystemen, die zum Teil zum Zeitpunkt des Erscheinens dieses Berichtes bereits erhältlich sein dürften.

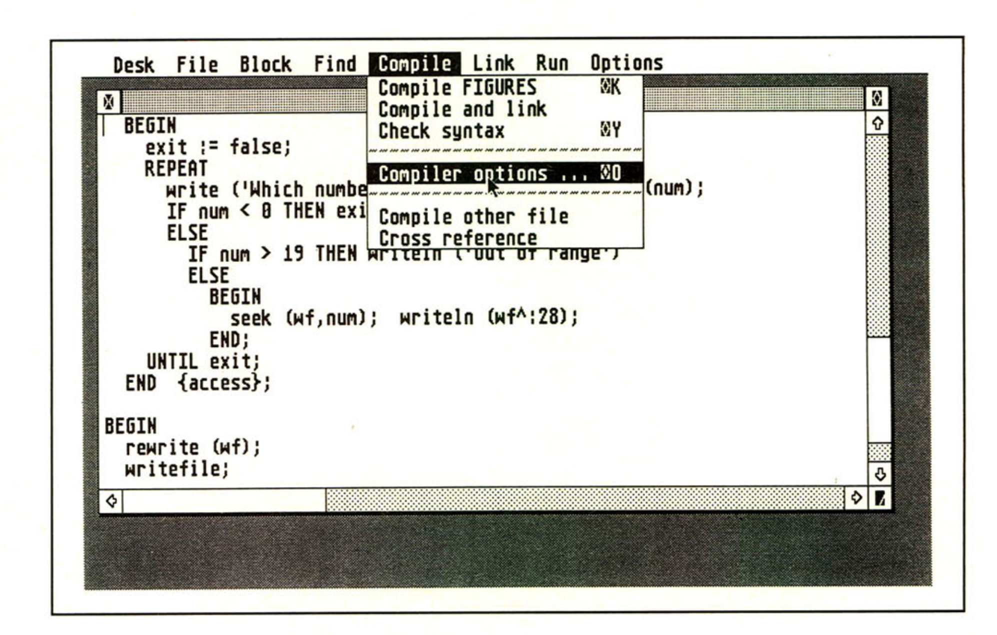


Bild 1: Die Wockbench von Prospero Pascal

Zuerst ein paar Gemeinsamkeiten: Beide Pascal-Systeme werden vollständig aus einer GEM-Shell bedient, das Verlassen dieser Shell dürfte während der Entwicklung eines Programmes in den seltensten Fällen notwendig sein. Bei Pascal Plus ist eine derartige Bedienerführung ja nichts neues, bereits in der älteren Version war eine Shell vorgesehen, auf den Komfort eines GEM-Texteditors mußten CCD-User aber bisher verzichten, es sei denn, sie waren stolze Besitzer des CCD-Editors TEMPUS, der seinen Beinamen 'DER Editor' nicht zu Unrecht trägt. Ein GEM-Editor mit allen notwendigen Funktionen (zwar nicht TEMPUS, aber auch nicht schlecht) ist in Pascal Plus jetzt enthalten, die Shell ist erheblich umfangreicher geworden.

Auch Prospero Pascal, dessen Handling bisher eher ein Denkmal der Vorzeit war, wurde völlig neu eingekleidet, und zwar in eine sehr funktionale Shell. Die Besonderheit dieser Shell ist, daß der Editor hier nicht als Overlay nachgeladen wird, wie es sich gebührt, sondern fest in die Shell integriert ist. Das ist hübsch, weil man einen Text compilieren kann, ohne ihn vorher abzuspeichern, einfach so, wie er im Rechner steht. Besonders schön ist dann allerdings, wenn man nach getaner Arbeit und dem Griff zum Ausschalter des Rechners feststellt, daß man sein Programm auch nach erfolgreicher Kompilierung nicht abgepeichert hat. Frohes Tippen. Die Shell gefällt aber ebenfalls sehr.

Lieferumfang

Beide Systeme sind im Grunde ähnlich aufgebaut: Geliefert werden Shell, Editor, Compiler und Linker. Zusätzlich enthält das CCD-Paket einen sehr leistungsfähigen und auch ausreichend schnellen Quelltextformatierer, der sehr übersichtliche und informative Listings liefert (z.B. reservierte Worte fettgedruckt, korrekt eingerückt, am Rande die Schachtelungstiefe der Zeile angegeben usw.). Prospero liefert einen Cross-Reference-Generator, der leider nicht der schnellste ist, sowie ein Library-Utility mit. Einige Hilfsbibliotheken (für GEM, TOS, Nützliches) sind ebenfalls beiden gemeinsam. Eine große Besonderheit, die sich jedoch in den illustren Kreisen der Compilerbauer immer mehr durchzusetzen

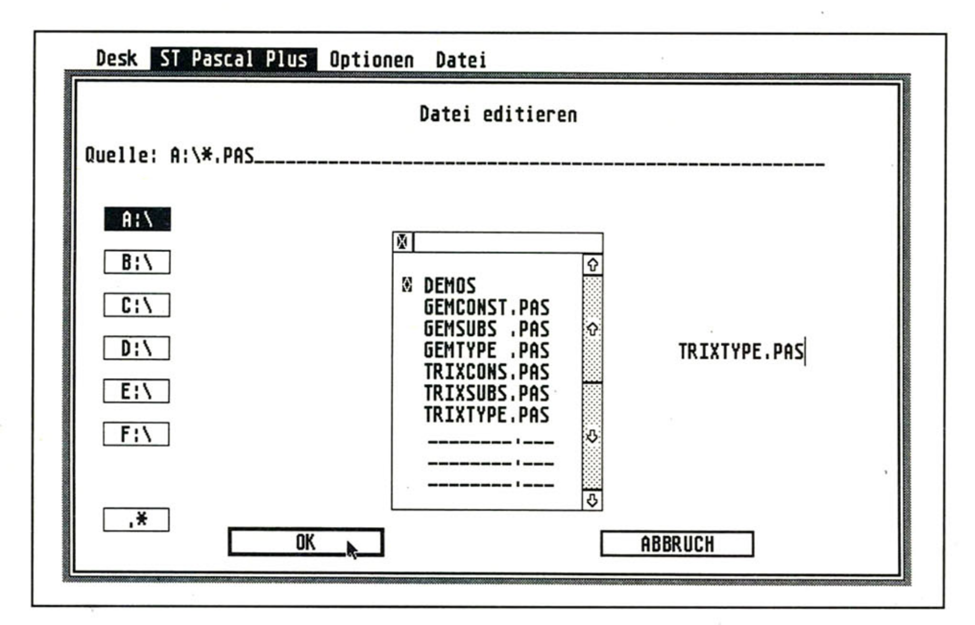


Bild 2: Die Fileselektorbox von ST Pascal Plus

beginnt, hat das ProPascal-System seiner deutschen Konkurrenz voraus: Einen Source-Level Debugger. Häh ? Nun, einige findige Entwickler auf größeren Rechnern und Workstations sagten sich vor geraumer Zeit, daß es doch eine Schande sei, über modernste Sprachen, Compiler usw. zu verfügen, aber bei der Fehlersuche nach obligatorischem Absturz auf die reine Intuition oder gar -igitt- auf Maschinensprache-Kenntnisse angewiesen zu sein. Auch Debugger wie SID, der ja zum C-Entwicklungspaket des Atari gehört, helfen nicht allzuviel, wenn es darum geht, lokale Variablen in einer zweieinhalbtausendmal rekursiv aufgerufenen Prozedur zu finden, oder auch einfach eine bestimmte Variable während der Ausführung des Programmes zu verändern - ohne erneuten Compilerlauf, versteht sich. Kurzum, herkömmliche Werkzeuge für die Fehlersuche lassen einen geplagten Programmierer immer dann im Stich, wenn er sie am nötigsten braucht, nämlich dann, wenn man die Möglichkeiten einer Hochsprache zur Strukturierung und ähnliche Annehmlichkeiten des modernen Programmierens nutzen will und Fehler in derartigen Gebilden auch halbwegs schnell beseitigen will. Um es kurz zu machen: So entstanden die ersten Source-Level Debugger, die eben genau die Fähigkeit besitzen, die ihr Name andeutet: In einem Testlauf eines kompilierten Programmes jederzeit Zugriff auf den Source-Text des gerade ausgeführten Programmes zu haben, Breakpunkte im Source-Code zu setzen, Variablen (auch rein lokal) tracen und ändern oder die Ausführung des Programmes an beliebiger Stelle fortsetzen zu können. Sie sehen also, ein guter Source-Level Debugger ist eine Art eierlegendes Wollmilchsau-Raubtier in der Welt der Wanzen.

Ein solcher Source-Level-Debugger, der über wirklich beachtenswerte Fähigkeiten verfügt, ist im ProPascal-System enthalten - und der Autor ist nach seinen ersten Erfahrungen mit diesem wohl ersten auf dem ST erhältlichen Muster seiner Art ganz hingerissen. Aber auf diesem Feld ist Konkurrenz bereits in Sicht: Die Version 2.0 des Pascal-Systems von Metacomco, das bei Erscheinen dieser Ausgabe ebenfalls erhältlich sein soll, verfügt ebenfalls über einen derartigen Debugger, über den man sich wahre Wunderdinge erzählt - nun, man wird sehen.

GEM-Einbindung

Der nächste Punkt der allgemeinen Ausführungen über die Herren Pascal betrifft die Einbindung der GEM-Aufrufe ins System. Hier scheiden sich die Geister ganz erheblich. Prospero hat sich für einen sehr eleganten Weg entschieden, nämlich den, sämtliche Datentypen, Konstanten und Prozeduraufrufe des Betriebssystem in Include- und Linkdateien zur Verfügung zu stellen, wodurch GEM-Programme aus anderen Sprachen, z.B. aus C, leicht nach Prospero Pascal (oder Fortran, das im letzten Heft getestet wurde und sich lediglich im Compiler von dem Pascal-System unterscheidet) übersetzt werden können bzw. umgekehrt. Die Implementierung der GEM-Aufrufe ist extrem geschickt gelöst, die unflexible Handhabung von Prozeduren versus Funktionen in Pascal ist elegant umgangen worden. Zusätzliche Funktionen, die nicht direkt in GEM enthalten sind, erleichtern manch unangenehme Arbeit. Leider hat man an einem Punkt, der allerdings alles andere als selbstverständlich ist, eine Möglichkeit versäumt: In die Objekt-Library wurde keine Routine einge-

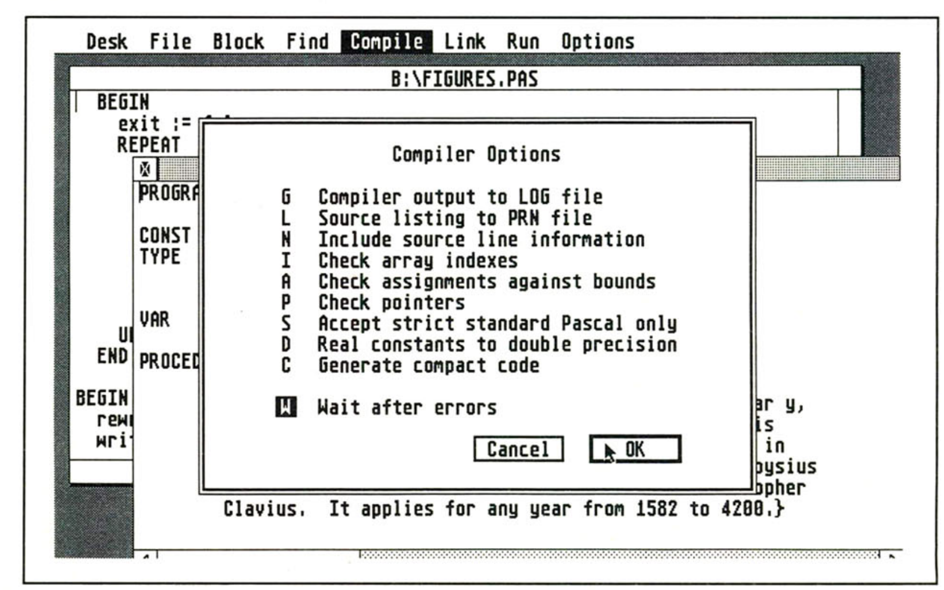


Bild 3: Die Compiler-Optionen von Prospero Pascal

fügt, die es ermöglicht, Pascal-Prozeduren zur Bearbeitung von 'Userdef'-Objektstrukturen zu verwenden, was wunderhübsch, die Krönung sozusagen, gewesen wäre, eben weil es in Pascal (in korrektem Pascal) völlig unmöglich ist, so etwas zu programmieren. Aber wie gesagt, das war eigentlich auch nicht zu erwarten, aber bei der Qualität der GEM-Bindings wird man schnell unverschämt. Ein dickes Lob für die Prospero GEM-Library.

CCD macht es dem an die Standard-Aufrufe gewöhnten Programmierer da erheblich schwerer. Benutzern älterer Pascal Plus-Versionen ist die GEM-Library des Systems bekannt, Newcomern sei berichtet, daß in der Library eine Auswahl von Funktionen des GEM vorhanden ist, die entweder einige Standard-GEM-Aufrufe unter neuem Namen zusammenfaßt oder für Verwirrung sorgt, indem sie nur einen einzigen Standard-GEM-Aufruf mit neuem Namen enthält oder gar, um das Durcheinander komplett zu machen, gar einzelne Aufrufe unter ihrem Original-Namen verwendet. Manche der so entstandenen Aufrufe machen es dem Neuling möglicherweise leichter, ans GEM heranzutreten, aber im allgemeinen möchte ich bestreiten, daß diese Art der GEM-Einbindung sehr nützlich ist, zumal, entgegen der anderslautenden Werbung, der Handbuchteil, der sich mit diesen Bindings beschäftigt, keineswegs sehr viel leichter verständlich und kürzer als viele der bisher erschienenen Bücher über die Standard-Aufrufe ist. Der Umstieg auf andere Sprachen oder die Verwendung von Hilfsroutinen für besondere Aufgaben, die man zum Beispiel in Zeitungen entdecken kann, wird meines Erachtens durch eine solche GEM-Einbindung unnötig erschwert. Außerdem sind, wie bereits erwähnt, nicht alle GEM-Funktionen enthalten, einige zusätzliche hinzugefügt usw. In der bisherigen Version des Pascal Plus Systems konnte man ja wenigstens völlig problemlos mit Hilfe der Dateien 'aesbind' und 'vdibind' die Originalfunktionen verwenden, vorrausgesetzt, man brachte den Fleiß auf, erst einmal die korrekten Procedurdefinitionen für die vielen, vielen GEM-Funktionen in seinen Computer einzutippen, da ein Include-File mit solchen Definitionen leider nie vorgesehen war. Mit der verbesserten GEM-Bibliothek der neuesten Pascal-Plus Version ist das nicht mehr ganz so problemlos: Zwar wurden viele Fehler aus der alten GEM-Library entfernt und viele Einschränkungen durch neu eingeführte Funktionen hinfällig, doch leider wurde im Zuge dieser Verbesserungen auch einiges verändert, so daß es nicht mehr so ohne weiteres möglich ist, einfach eine GEM-Prozedur zu definieren und sie dann durch linken mit 'aesbind' oder 'vdibind' auch tatsächlich nutzen zu können. Nein, etwas Mühe gehört dazu; einige Spezialvariablen, die beim alten System automatisch definiert waren, sind jetzt in die Include-Dateien, die zur CCD-GEM-Library gehören, abgewandert, so daß man diese mitkompilieren, außerdem aber auch die GEM-Library von CCD mitlinken muß, um ein Programm mit Standard-GEM lauffähig zu machen, was die ganze Sache etwas unbequem macht. Dummerweise tragen einige Prozedur-Aufrufe in der CCD-Library in der neuen Version aber doch wieder Standard-Namen, was dazu führt, das unter Umständen auch noch einiges an Änderungen fällig wird, damit ein älteres Programm fehlerfrei kompiliert und gelinkt werden kann. Mir erscheint diese Art der GEM-Einbindung recht ungeschickt, nichts

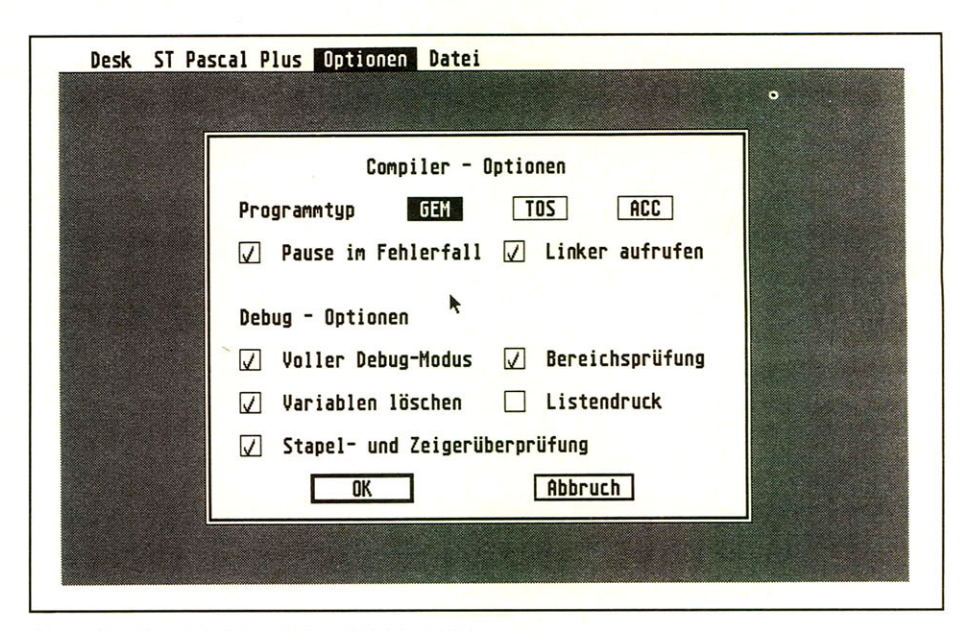


Bild 4: Die Compiler-Optionen von ST Pascal Plus

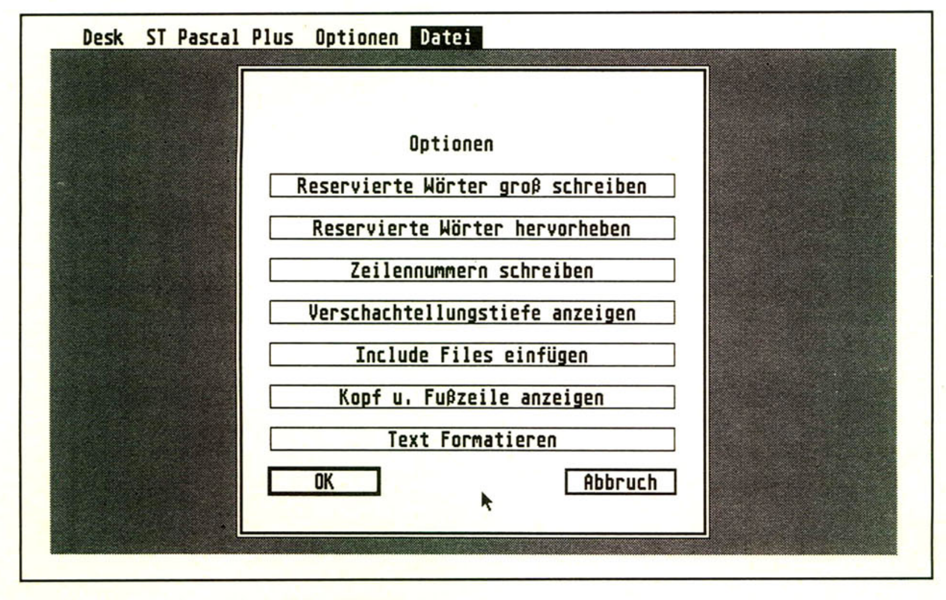


Bild 5: Formatierte Ausgabemöglichkeit bei ST Pascal Plus

gegen Erleichterungsversuche für geplagte Anfänger, aber bitte doch mit Wahrung der Kompatibilität, ohne den Standard-gewöhnten durch große Mühen zu verschrecken oder quasi zur Benutzung eines uneleganten Bindings zu zwingen.

Sehr gelungen ist dagegen eine Bibliothek namens PASTRIX (welch hübscher Name...), die wirklich äußerst nützliche Funktionen enthält, die es z.B. ermöglichen, ganze Bildschirminhalte in einer Variable abzuspeichern, zu peeken und poken, Adressen von Variablen zu finden, Datum und Zeit und Strings zu konvertieren usw., usw. Einige Probleme, die die strenge Pascal-Typüberprüfung besonders bei der systemnahen oder GEM-Programmierung bietet, lassen sich mit Hilfe dieser Funktionen recht einfach umgehen.

Bei Prospero sind derartige Funktionen bereits im Sprachumfang enthalten (Peek, Poke usw.) bzw. in die TOS-Bibliothek integriert (Zeit-/Datums-Konvertierung).

Dokumentation

Das CCD-Pascal wird selbstverständlich mit einem deutschen Handbuch geliefert. Die gesamte Dokumentation besteht aus dem Original-Handbuch der letzten Pascal-Plus-Version sowie einem dünnen Ergänzungsband, der die Erweiterungen und Änderungen der neuen Version beschreibt. Die Anleitung ist gut brauchbar, aber nicht immer optimal übersichtlich.

Für ProPascal gibt es, wie könnte es auch anders sein, nur eine englische Anleitung. Drei Bände sind es, einer für den Compiler, die TOS- und Hilfsbibliotheken, sowie jeweils einer für die VDI- und AES-Bindings. Die Anleitung ist gut und leicht verständlich geschrieben; ärgerlich in der AES-Anleitung ist, daß dort auch alle Funktionen der GEM-Version 2.0 (unter MS-DOS) enthalten sind, was an sich nicht weiter schlimm wäre, wenn man nicht vergessen hätte, irgendwo aufzulisten, was das GEM im Atari nun eigentlich kann und was nicht. Etwas ungeschickt ist auch der Compiler-Band, der sich aus mehreren Teilen zusammensetzt, die leider kein gemeinsames Register haben, so daß man, wenn man nicht weiß, wo genau man etwas suchen soll, immer etwas blättern muß.

	ProPascal	Pascal Plus
Floattest 500.000 Multiplikationen; 50.000 Sinus; 25.000 Matrix*Vektor (4*4-Matrix)	C:15 sec. L: 17 sec. 1 externe Library, 20 KByte Multiplikation: 1 Minute 15 Sekunden, Sinus: 1Minute 30 Sekunden Matrix/Vektor: 1Minute 23 Sekunden	C: 5 sec. L: 5 sec. 1 externe Library, 11 KByte 1:50 3:15 2:05
Tower Türme von Hanoi	C: 10 sec. L: 17 sec. 1 externe Library, 20K 17 Scheiben: 29 Sekunden	C: 4 sec. L: 3 sec. 1 externe Library, 6K 17 Scheiben: 20 Sekunden
Sieve Primzahlen-Sieb	C: 11 sec. L: 17 sec. 1 externe Library, 20 Kbyte 10 Durchlufe bis 1001, mit Ausgabe: 9 Sekunden	C: 5 sec. L: 3 sec. 1 externe Library, 6 K 11 Sekunden
Benchmark-Ergebnisse: Mathematik mit normaler Genauigkeit. ProPascal Linker ohne Selectiv-Scan-Option für optimale Code-Länge.		

Tabelle: Pascal Benchmarks

Spracherweiterungen

Beide Pascal-Compiler sind in einigen Punkten gegenüber der Norm erweitert. Fast schon Standard ist der Stringtyp samt den dazugehörigen Funktionen zu seiner Manipulation. In ProPascal sind Strings voll dynamisch, das heißt, ihre Länge (bis zu 32KByte) muß nicht festgelegt werden.

Beide Compiler erlauben auch modulares Kompilieren. Dabei können globale Variablen von allen Programmteilen genutzt werden, aber auch geschützte Variablenbereiche sind möglich.

In Sachen Zahlen: Der Prospero-Compiler versteht verschiedene Real-Formate, nämlich mit 7 und 16 Stellen Genauigkeit, sowie 32-Bit-Integer-Zahlen. Vordefiniert sind Subranges mit 8 und 16 Bit Größe mit und ohne Vorzeichen.

Ein genereller Pointer-Typ ist vorhanden, der mit jedem beliebigen Pointer kompatibel ist.

Bei CCD gibt es ein Real-Format, das 11 Stellen Genauigkeit bietet. Integer-Zahlen sind 16 Bit, vordefiniert ist der Unterbereich Byte, Long_Integer-Zahlen sind für 32 Bit-Werte vorgesehen.

Weitere gängige Erweiterungen wie 'Else' in 'CASE'-Anweisungen, Random Access-Files und bitweise logische Operationen sind ebenfalls in beiden Compilern enthalten.

Eine hochinteressante Erweiterung bietet nur ProPascal: Eine 'ownerr'-Routine erlaubt es, Run-Time-Fehler, wie z.B. Division durch Null, an eine eigene Fehlerbehandlungsroutine zu übergeben. Auf diese Weise kann man Programme sehr sicher gegen Fehler machen.

Der CCD-Compiler erlaubt noch etwas, das eigentlich der Pascal-Philosophie strikt entgegengeht: Variablen dürfen, wie in Basic, an jeder beliebigen Stelle des Programms definiert werden und sind dann nur allen später definierten Teilen des Programmes bekannt. Eine besonders glückliche Idee scheint mir das nicht zu sein, aber CCD hat diese Möglichkeit laut Handbuch auf Wunsch zahlreicher Kunden implementiert.

Praxis

Mit beiden Systemen, dem Manager von CCD und der Prospero-Workbench, kann man vernünftig und schnell arbeiten. Mir persönlich gefällt die Prospero-Workbench etwas besser, weil bei ihr die Ladezeiten für den Editor entfallen. Dafür ist der CCD-Editor etwas besser, denn der ProPascal-Editor hat ein kleines Pro-

blem, das man von den ersten 1st_ Word-Versionen kennt: Der Cursor läuft nach. Sehr nützlich bei der CCD-Shell sind die Kopier-kommandos, so daß man die Shell nicht dauernd verlassen muß. In der Praxis ist das Pascal Plus-Paket wegen der kürzeren Turn-around-Zeiten immer dann im Vorteil, wenn keine schwierigeren Fehler zu suchen sind. Durch den integrierten Source-Level-Debugger macht ProPascal die Fehlersuche extrem angenehm, was manchen Compiler-Zyklus spart. Auf der Disk braucht der CCD-Compiler weniger Platz, weil er ohne Zwischenfiles auskommt. ProPascal erzeugt eines.

Der Compiler-Output von CCD ist kompatibel zu anderen Sprachen, oder besser gesagt, der CCD-Linker versteht das Digital-Research-Linkformat.

Prospero benutzt, wie schon in der alten Version, das Objektcode-Format von GST.

Beim Testen der Compiler entstand bei mir immer stärker der Wunsch, einen Compiler zu besitzen, der die guten Eigenschaften beider Systeme in sich vereinigt. Beide Compiler haben nützliche Bibliotheken, praktische Verbesserungen gegenüber der Pascal-Norm, es erlauben z.B. beide modulare Kompilierung. Ich hätte gerne die Erweiterungen beider.

Beide Shells haben einiges für sich. Besonders deutlich wird der Wunsch aber bei den Benchmark-Tests: Während der CCD-Compiler wirklich in Windeseile arbeitet (er ist in der neuen Version noch schneller geworden, als er sowieso schon war), also für extrem niedrige Turn-around Zeiten sorgt, ist das ProPascal eher etwas gemächlich. Leider gilt für die Ergebnisse der Entwicklung, vor allem im Bereich Fließkommarech-nung, genau das Gegenteil: Der CCD-Compiler hat hier seine deutliche Achillesferse. Den kompakteren Code erzeugt nun wiederum das CCD-System, die Fehlersuche bei ProPascal ist durch Probe, wie der Source-Level-Debugger sich zu nennen beliebt, natürlich ganz erheblich vereinfacht.

Zur Dokumentation einige Benchmarks im einzelnen (siehe Tabelle).

Zusammenfassung: Dieser Compiler-Vergleich ist irgendwie frustrierend. Beide Compiler haben in bestimmten Gebieten ganz besondere Vorzüge. Im Handling erweist sich der CCD-Compiler wegen seiner Geschwindigkeit als etwas angenehmer, mir persönlich gefällt das Arbeiten mit der Prospero-Shell etwas besser, aber der Compiler ist leider ein bißchen langsam. Debuggen ist natürlich die Stärke von ProPascal, was bei komplexen Programmen den Nachteil der niedrigen Geschwindigkeit durchaus aufwiegen kann. Beim Arbeiten mit GEM ist die Bibliothek des ProPascal-Systems klarer Sieger. Bei den fertigen Programmen dreht sich das Bild: Prosperos Pascal ist plötzlich viel schneller, die Programme sind allerdings um einiges länger, weil anscheinend grund-sätzlich ein aufwendigeres RunTime-Paket dazugelinkt wird (je länger die Programme, desto weniger fällt dieser Anteil ins Gewicht...). CCD Pascal rechnet arg langsam, eine Version für den Arithmetik-Coprozessor soll da demnächst Abhilfe schaffen. Aber auch außerhalb der Arithmetik ist CCD-Pascal langsamer in der Ausführung, wie die Benchmarks erweisen.

Das Preis-/Leistungs-Verhältnis der Pakete ist in Ordnung.

Kurzum: Es soll jeder nach seiner Fasson selig werden, mir gefallen die Pascals beide sehr gut, weil man mit beiden Systemen vernünftig und professionell arbeiten kann. Trotzdem bin ich sehr gespannt auf das, was die Konkurrenz, allen voran Metacomco, in Kürze auf dem Pascal-Sektor zu bieten hat. Vielleicht findet sich ja doch noch das Pascal meiner Wunschträume, das frei von Fehl und Tadel mir nur zur Freude dienet...

CS

```
74   1    out_char(#27); out_char('f'); { cursor off    }
75   1   { zeichnen }
76   1   REPEAT
77    2    FOR index := 0 TO max_draw_lines DO
78   2    BEGIN
79   3     FOR i := 0 TO 3 DO
80   3     BEGIN
81   4       t := ball[i] + vel[i];
82   4      IF t >= max[i]
83   4      THEN
84   4      BEGIN
85   5        t := max[i] * 2 - ball[i] - vel[i];
86   5       vel[i] := -vel[i]
87   4      END;
88    4       IF odd(i) THEN min := min_y ELSE min := min_x;
89   4      IF t < min
90   4      THEN
91   4      BEGIN
92   5        IF t < 0 THEN t := -t ELSE t := min;
93   5       vel[i] := -vel[i]
```

Beispiel einer formatierten Ausgabe mit Pascal Plus

ENDE

BILDERSPIELE TEIL 5

Mit den bisher dargestellten Mitteln können Sie einfache Strichzeichnungen auf den Bildschirm eines Computers bringen. So besonders hübsch ist das aber noch nicht. Etwas mehr gibt es doch zu tun. Werfen wir also einen Blick auf die Schritte, die hinführen zur künstlichen Wirklichkeit aus dem Computer...

Eine Strichzeichnung in bisheriger Form ist aus verschiedenen Gründen nicht besonders wirklichkeitsnah. Der erste und deutlichste Grund ist der, daß die 'Wirklichkeit' nicht aus lauter dünnen Strichen besteht, sondern aus Objekten, die meist ziemlich massiv (Spinnen weghören) und auch nur in den seltensten Fällen durchsichtig sind.
Nehmen wir an, dieses Problem sei gelöst. Wir haben massive Oberflächen. Jetzt stellt sich die Frage, warum natürliche Oberflächen aussehen, wie sie es eben nun mal tun. Die Antwort auf diese Frage heißt zuerst einmal 'Beleuchtung'. Die Art des Lichtes, das auf ein Objekt fällt und von diesem reflektiert wird, ist entscheidend für sein Aussehen. Na gut, seien wir ehrlich, auch dies genügt nicht, man muß sagen: Die Art des Lichtes, das auf ein Objekt fällt, zusammen mit der Art, wie das Objekt Licht reflektiert, entscheidet über sein Aussehen.
Ein Objekt ? Wo Licht ist, ist auch Schatten. Sobald es zwei Objekte gibt, besteht die Chance, daß man den Schatten des einen Objektes auf dem anderen Objekt sieht. Sehr unangenehm und auch nicht leicht zu lösen, dieses Problem.
Nehmen wir einmal an, ein Objekt reflektiert nicht nur Licht sondern läßt auch noch Licht hindurch, sei also transparent. Hier spielen, mehr noch als bei der Reflektion, Materialeinflüsse eine große Rolle, schließlich werden Lichtstrahlen im allgemeinen beim Übergang vom einen in ein anderes Medium gebrochen.
Daran, wie oft in den letzten Sätzen das Wort Licht gefallen ist, sehen Sie, wie wichtig dieser Punkt in Zukunft sein wird. Aber darüber machen wir uns später Gedanken. Zuerst einmal sollen uns die massiven Oberflächen interessieren.
Die Probleme mit guten Realismus-Effekten, die die herkömmlichen Verfahren bieten, werden uns dann zum Ende dieser Serie zu einem Verfahren führen, das mit großer Einfachheit und Eleganz, leider aber extremem Bedarf an Rechenzeit, fast alle Probleme herkömmlicher Verfahren zu beheben vermag, und das deshalb für professionelle Computergrafik heute unverzichtbar geworden ist. Es hört auf den bezeichnenden Namen Ray Tracing. Aber dazu an anderer Stelle mehr...
Wie kann man ein massives Objekt modellieren ? Dieses Problem ist uns nicht neu, wir haben darüber bereits nachgedacht und uns der Einfachheit halber dafür entschieden, es aus Flächenstücken, die aneinandergesetzt sind, aufzubauen. Ein Würfel zum Beispiel besteht dann aus... Na ? ...Richtig, 6 Flächen, 12 Linien und 8 Punkten. Wir müssen also nur unseren Objektdatentyp um eine Fläche erweitern, die aus einer Reihe von Linien zusammengesetzt ist. Das war nicht schwer. Nun stellt sich allerdings die Frage, wie man dem Computer klarmacht, welche Flächen eigentlich von einem beliebigen Blickpunkt aus sichtbar sind.
Dieses Problem beschäftigt die Programmierer von Grafiksystemen schon lange, und, um das gleich zu Beginn zu sagen, eine optimale Lösung gibt es nicht. Leider hat hier jeder bekannte Algorithmus einen Nachteil: entweder er ist langsam oder er ist schlecht, er braucht riesige Speichermengen (1000*1000 Pixel z.B. 7 MByte) oder ist nur für sehr spezielle Anwendungen brauchbar. Aus der ungeheuren Vielzahl der Möglichkeiten sollen in dieser Serie zwei Algorithmen, die in der Praxis besonders häufig zur Anwendung kommen, vorgestellt werden.
Erwähnt sei zu der Vielzahl der Hidden-Surface-Algorithmen noch, daß auch das oben angesprochene Ray Tracing ursprünglich nur für diesen Zweck erdacht wurde. Welche Möglichkeiten in diesem Verfahren stecken, entdeckte man erst im Zuge der Beschäftigung mit Beleuchtungsmodellen usw. langsam Schritt für Schritt. Sie sehen, ich will Sie neugierig machen.
Diese Hidden-Surface-Routinen werden nicht mehr implementiert werden. Im Rahmen dieser Serie wäre eine vollständige Implementierung einfach zu umfangreich und in verständlicher Form auch zu wenig

effizient. Ich möchte daher versuchen, Sie in die Lage zu versetzen, mit möglichst wenig Aufwand selbst entsprechende Routinen zu schreiben.

Zum Warmwerden ein etwas vereinfachtes Problem. Wir beschränken uns auf ein harmloses Objekt, z.B. den berühmten Würfel. Welche Teile sind von schräg oben sichtbar, welche nicht ?

Stellen Sie sich einmal ein Viereck vor, dessen Ecken Sie streng im Uhrzeigersinn der Reihe nach numeriert haben. Jetzt drehen Sie das Viereck auf den Rücken. Fällt Ihnen etwas auf ? Jawohl, die Reihenfolge der Numerierung ändert sich. Wenn Sie ein im Uhrzeigersinn numeriertes Viereck umdrehen, ändert sich der Drehsinn der Numerierung. Diese kleine Nettigkeit werden wir verwenden, um dem Computer die Arbeit ein wenig zu erleichtern. Wie das ? Sie legen sich die sechs Flächen auf den Tisch, aus denen der Würfel besteht und numerieren ihre Ecken in der gleichen Reihenfolge durch, immer auf der gleichen Seite. Jetzt kleben Sie den Würfel so zusammen, daß die numerierte Seite immer außen ist. Wenn Sie den fertigen Würfel von einer Seite aus anschauen und Sie ihn aus transparentem Material gebaut haben, werden Sie sehen, daß die Rückseite jetzt einen anderen Drehsinn der Numerierung zeigt als die Vorderseite. Wenn Sie alle Seiten im Uhrzeigersinn numeriert haben, ist die Rückseite von Ihrem Blickpunkt aus jetzt entgegen dem Uhrzeigersinn numeriert. Bild 1 zur Illustration.

Wie bringt man das dem Computer bei ? Ganz einfach. Sie erinnern sich an das Skalarprodukt zweier Vektoren ? Betrag von Vektor 1 mal Betrag von Vektor 2 mal Cosinus des eingeschlossenen Winkels. Wunderbar. Sie kennen den Vektor, entlang dem Sie den Würfel betrachten (Vom Aug- zum Blickpunkt). Sie werden in Kürze einen Vektor kennen, der auf einer Fläche senkrecht steht, und dessen Richtung vom Drehsinn der Eckpunktnumerierung abhängt. Der senkrechte Vektor ist natürlich die Ebenennormale einer Seite, deren Richtung vom Drehsinn abhängt. Doch dazu später mehr, nehmen wir jetzt einfach mal an, wir kennten einen Vektor, der auf der Ebene senkrecht steht, und zwar so, daß er die Fläche immer auf der Seite verläßt, auf die Sie vorhin die Nummern geschrieben haben. Schauen Sie sich Bild 2 an. Das Vorzeichen des Skalarprodukts hängt nur vom Cosinus ab. Wenn der Cosinus positiv ist, ist es auch das Skalarprodukt. Und umgekehrt, wie Sie aus der letzen Folge wissen. Das Skalarprodukt ist, wie sich einfach aus dem Cosinus ergibt, immer dann positiv, wenn die beiden Vektoren in einem Winkel zwischen 270 und 90 Grad aufeinanderstehen. Bild 3 zeigt die Bedeutung dieser Aussage: Wenn der Sichtvektor und der Normalenvektor der Fläche wie oben definiert, aufeinanderzuzeigen, dann ist die Fläche eine Vorderseite und somit sichtbar (Vom vorgegebenen Blickpunkt aus). Andernfalls ist sie eben unsichtbar, weil sie eine Rückseite des Objektes ist, und Rückseiten kann man nicht sehen.

Das funktioniert so natürlich nur bei konvexen Körpern und dann, wenn Sie die Oberflächen alle im gleichen Drehsinn numerieren. Aber zurück zu der Oberflächennormalen. Sie erinnern sich an das Kreuzprodukt zweier Vektoren. Das Ergebnis ist ein Vektor, der auf beiden Vektoren senkrecht steht, also auf der von Ihnen aufgespannten Ebene. Bild 1 zeigt, daß sich die Richtung des Kreuzvektors aus dem Drehsinn ergibt, wenn man bei allen Polygonen der Fläche nach der Objekttransformation mit dem vom Sichtpunkt aus gleichen Punkt beginnend aus zwei Vektoren den Normalenvektor bildet. Man sucht sich also einen Punkt aus, zum Beispiel den am weitesten oben liegenden Punkt. Dann nimmt man den Vorgängerpunkt und errechnet aus beiden Punkten einen Seitenvektor. Das gleiche tut man mit dem folgenden Punkt. Bevor man daraus jetzt ein Kreuzprodukt bildet, sollte man feststellen, ob die Punkte auf einer Linie liegen, denn sonst ist das Kreuzprodukt gleich 0, was nicht sehr sinnvolle Folgen für das Skalarprodukt mit dem Sichtbarkeitsvektor hätte. Wenn alle Punkte der Fläche auf einer Linie liegen, sieht man sie genau von der Seite und kann die Fläche ignorieren, als sei sie unsichtbar. Wenn die Punkte nicht auf einer Linie liegen, wird also das Kreuzprodukt der zwei errechneten Vektoren gebildet und dann das Skalarprodukt mit dem Sichtvektor. Ist dieses kleiner 0, ist die Fläche sichtbar, andernfalls nicht. Dieser Test muß natürlich für jedes Polygon eines Objektes

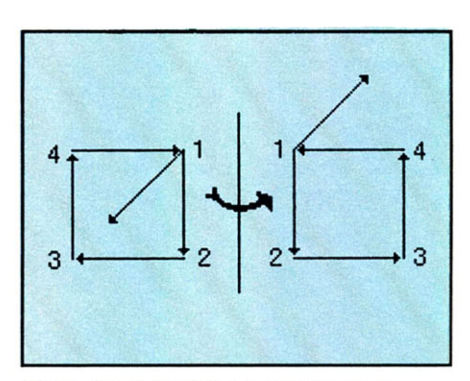


Bild 1 : Die Reihenfolge der Punkte eines Polygons definiert den Drehsinn und die Richtung des Normalenvektors

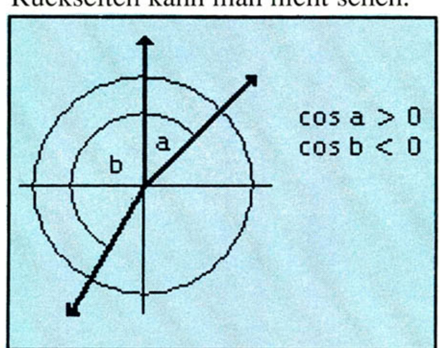


Bild 2: Das Vorzeichen des Cosinus ist vom eingeschlossenen Winkel bestimmt.

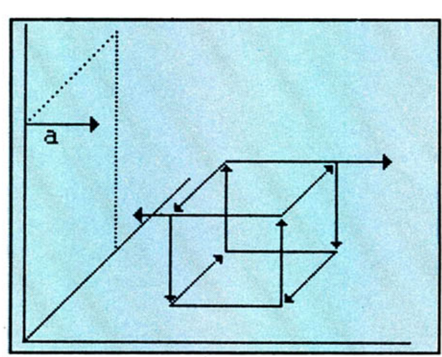


Bild 3 . Die Seiten des Würfels wurden im Uhrzeigersinn numeriert. Daher zeigt die Normale der Rückseite jetzt in die entgegengesetzte Richtung. Aus dem Sichtvektor a der Bildebene ergibt sich die Sichtbarkeit jeder Fläche.

durchgeführt werden. Das ganze Ablaufdiagramm für diesen 'Backface-Removal' genannten Vorgang finden Sie in Bild 4.
Falls Sie ein Grafiksystem vereinfachen wollen und dem Hidden-Surface-Algorithmus einige Arbeit sparen wollen, lohnt es sich, eine Backface-Removal-Routine vor die eigentlichen Hidden-Surface-Tests zu setzen. Die Anzahl der zu testenden Flächen wird nämlich radikal reduziert. Bei mehreren Objekten ist eine solche Routine alleine natürlich deshalb nicht ausreichend, weil sie nicht feststellen kann, ob eine dem Betrachter zugewandte Oberfläche von einer ebenfalls dem Betrachter zugewandten Oberfläche eines anderen Objekts verdeckt wird.
Das Verfahren setzt voraus, daß das Objekt bereits entsprechend dem Blickpunkt transformiert wurde, es muß allerdings vor der Projektion verwendet werden (Natürlich).

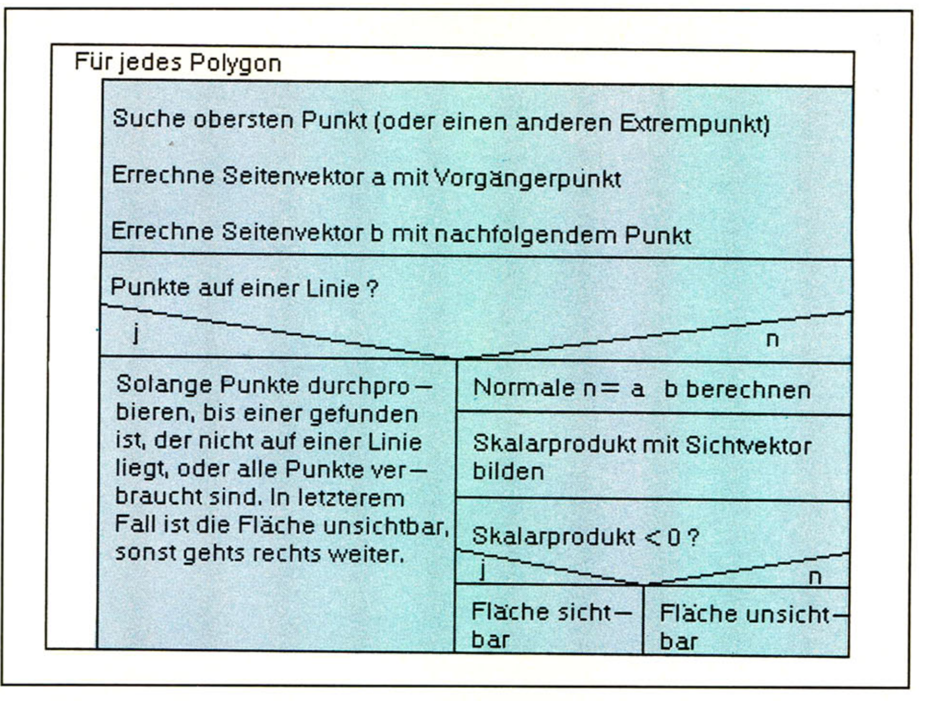


Bild 4: Struktogramm für Backface-Removal. Dieser Algorithmus, der so nur für konvexe Formen gilt, erleichtert einem nachgeschalteten Hidden Surface-Prozessor die Arbeit.

Hidden-Surface-Algorithmen

Es gibt grundsätzlich zwei Klassen von Hidden-Surface-Routinen: Solche, die die auszublendenden Oberflächen mit Hilfe einer Sortierung ermitteln und solche die es ohne Sortierung tun. Ich habe für beide Klassen einen vielverwendeten Algorithmus ausgesucht.

Zum Aufwärmen: z-Buffer

Der erste nennt sich 'z-Buffer-Algorithmus' und hat den Vorteil extrem einfach zu sein, vielleicht sogar der einfachste aller Hidden-Surface-Algorithmen. Gleichzeitig ist er schnell und kann leicht für Oberflächenschattierungen, also einfache Lichteffekte auf Oberflächen, erweitert werden. Hauptnachteil des Verfahrens ist der wahnsinnige Speicherbedarf. Im Zuge des Speicherpreisverfalls ist dieser Punkt allerdings nicht mehr so kritisch. Unangenehmer ist die Schwierigkeit, Transparenzeffekte und Bildqualitätsverbesserung mit Hilfe von Antialiasing (sorgt dafür, daß man die Computertreppchen in den Linien nicht so sieht) zu implementieren.
Der z-Buffer-Algorithmus ist unabhängig von der Komplexität der Szene, soweit es die Qualität und Fehlerhäufigkeit der Ergebnisse betrifft. Egal, wie wild Sie die Polygone im Raum verteilen - so einfach der z-Buffer zu programmieren ist, er verarbeitet alles, was viele seiner komplexeren Kollegen nicht von sich behaupten können. Wie bei allen derartigen Verfahren leidet natürlich die Effizienz - je mehr Polygone, desto länger dauert's.
Das Verfahren arbeitet im Bildschirmkoordinatensystem, da ja Informationen, die auf die einzelnen Bildpunkte des Bildschirms bezogen sind, verarbeitet werden. Allerdings muß bei der Projektion die z-Koordinate jedes Punktes zwischengespeichert werden, da sonst ja keine Tiefeninformation mehr vorliegt, nach der man über Sichtbarkeit entscheiden könnte.

Wie funktioniert's ?

Der Algorithmus bearbeitet jedes Polygon, also jede Fläche, einzeln; die Reihenfolge, in der die Flächen geliefert werden, spielt keine Rolle. Zu Beginn baut der Algorithmus einen großen Buffer auf, der in x- und y-Richtung die Größe des Bildschirms hat. In diesem Buffer wird jetzt für jedes bearbeitete Pixel eine Tiefeninformation gespeichert, wobei man im allgemeinen 20 Bit Tiefe für ausreichend hält. Bei einem Bildschirm mit 1000*1000 Pixeln ist das aber schon ein Speicherbedarf von 6 Megabyte, allerdings kann man auf Kosten der Effizienz auch Speicher sparen, indem man immer nur ein Viertel oder die Hälfte des Bildschirms auf einmal bearbeitet (Na, wie groß muß ein z-Buffer für einen Bildschirm mit ATARI-Auflösung sein ???). Initialisiert wird dieser Buffer auf die größtmögliche Tiefe, nämlich unendlich, entsprechend dem Wert 0 oder auch $FFFFF, ganz nach Geschmack.
Für jedes Pixel, das innerhalb des Polygons liegt, wird jetzt aus den Koordinaten der Eckpunkte die z-Koordinate berechnet. Dafür gibt es verschiedene Möglichkeiten.
Dann wird der errechnete Wert mit dem im z-Buffer befindlichen Wert für die entsprechende Pixelposition verglichen. Liegt der errechnete Wert näher beim Betrachter, wird das Pixel auf den Bildschirm gemalt, mit der diesem Polygon zugeordneten Farbe. Außerdem wird der z-Buffer-Wert durch den neuen Wert ersetzt. Ist der z-Wert des Pixels weiter weg als der

letzte z-Buffer-Wert passiert einfach gar nichts.
Ein einfaches Verfahren, nicht?
Struktogrammsüchtige finden den Ablauf etwas formaler gestaltet in Bild 5. Etwas anschaulicher kann man das ganze vielleicht so erklären: Zuerst stellt man fest, welchen Teil (also welche Pixel) des Bildschirms ein Polygon bedeckt. Dann errechnet man für jedes dieser Pixel die Entfernung zum Betrachter (Die Lage des Polygons im Raum ist ja bekannt, die z-Koordinate der Eckpunkte wurde bei der Transformation zwischengespeichert). Wenn noch kein Punkt gefunden wurde, der an der entsprechenden Pixel-Position näher beim Betrachter liegt (was im z-Buffer notiert wäre), wird das Pixel auf dem Bildschirm mit den Polygonattributen (Farbe, Helligkeit o.ä) gezeichnet, andernfalls passiert überhaupt nichts. Damit bei der Bearbeitung des nächsten Polygons nichts schiefgeht, muß natürlich die Tiefe eines gezeichneten Punktes im z-Buffer notiert werden, damit sichergestellt ist, daß nur näher beim Betrachter liegende Punkte einen gezeichneten Punkt übermalen dürfen.
Der z-Buffer ist also nichts weiter als ein Notizblock, auf dem für jeden Bildschirmpunkt notiert wird, wie weit entfernt ein eventuell an dieser Stelle des Bildschirms gezeichnetes Objekt vom Betrachter entfernt ist. Nur wenn ein von dem Objekt besetztes Pixel näher ist, als die im z-Buffer gespeicherte Entfernung, ist es sichtbar, wenn es weiter entfernt ist, wird es einfach übermalt. Die Objekte müssen also vor dem Zeichnen nicht sortiert werden, es ist völlig egal, in welcher Reihenfolge Polygone bei einem z-Buffer bearbeitet werden. Für jedes Pixel merkt der Algorithmus sich, wie groß die aktuelle Mindestentfernung vom Betrachter ist, ab der ein Pixel gezeichnet wird, bzw. verdeckt ist. Deshalb muß am Anfang der z-Buffer auch auf einen festen Wert initialisiert werden, der die maximal mögliche Entfernung angibt. Punkte des ersten Polygons, das bearbeitet wird, liegen dann auf jeden Fall näher, als jeder bisherige Punkt.

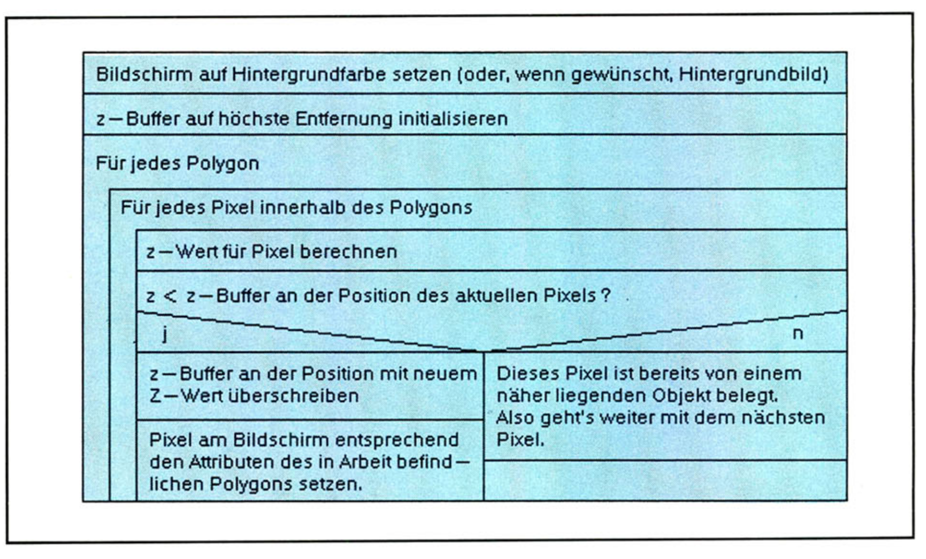


Bild 5: Der z-Buffer Algorithmus als Struktogramm.

Beim nächsten Polygon könnten schon Teile übergangen werden, die weiter entfernt liegen, als das erste bearbeitete Polygon.
Hier zeigt sich auch ein Vorteil, der sich aus dem 3D-Clipping ergibt. Die Tiefe des z-Buffers ist leichter sinnvoll auf einen bestimmten Wert zu begrenzen, wenn man die Tiefe 'unendlich' für den Ausschnitt der 'Welt', der gezeigt werden soll, ausschließen kann. Nach dem 3D-Clipping ist ja nur noch ein in der Tiefe beschränkter und bekannter Teil der 'Welt' übrig.
Hier kann man dann für die Tiefeninformation im z-Buffer auf Fließkommazahlen verzichten; eine Fließkommazahl, die üblicherweise mindestens 7 Stellen Genauigkeit aufweist, benötigt mindestens 4 Byte, was den z-Buffer doch etwas unhandlich groß macht. Wenn die x- und y-Genauigkeit im Bereich von 1000 Schritten (Bedenken Sie: Unsere 'Welt' befindet sich ja in dem Stadium der Darstellung, in dem der z-Buffer zur Anwendung kommt, bereits in der zu Bildschirmkoordinaten transformierten Form !) in jeder Richtung liegt, muß man ja nicht in z-Richtung 10^{38} Schritte, jeweils mit 6 Stellen Genauigkeit vorsehen. Üblicherweise geht man davon aus, daß bei der genannten Bildauflösung eine Tiefengenauigkeit von 1 bis 16 Millionen Schritten völlig ausreichend ist, entsprechend 20 - 24 Bit. Sichtbare Fehler durch falsch errechnete Tiefeninformation (Rundungsfehler müssen ja schließlich vermieden bzw, berücksichtigt werden) treten bei dieser Auflösung dann nicht mehr auf.
Als Zahlenformat wird meist Festkommadarstellung verwendet, da kaum ein System von Haus aus über ein Zahlenformat mit 20 oder 24 Bit Größe verfügt (meist nur 16 oder 32 Bit). Integer, oder in diesem Fall Long_Integer-Darstellung tut's natürlich auch, beide Darstellungen haben gegenüber dem Fließkommaformat auch noch den nicht unbeträchtlichen Vorteil, daß sich damit erheblich schneller rechnen läßt.
Schließlich muß noch das Problem betrachtet werden, wie man die Entfernung eines beliebigen in einem Polygon enthaltenen Punkts vom Betrachter errechnet. Eine einfache und komfortable Lösung sei hier kurz skizziert; weil sie aber so einfach ist, nicht im Detail ausgeführt:
Polygone liegen in einer Ebene. Ein Polygon besteht aus mindestens drei Punkten, was es möglich macht, eine eindeutige Ebenengleichung für es aufzustellen bzw. zu errechnen. Damit lassen sich aber die Koordinaten jedes einzelnen Punktes in der Ebene, also auch innerhalb des Polygons berechnen. Was will das Herz mehr?

ATARI ST SOFTWARE

EXKLUSIV

Multi-Tasking-C-Shell



- Multi-Tasking Betrieb
- Multi User Betrieb (2. Terminal an RS-232)
- Electronic Mail
- Beseitigt TOS Fehler

*** 298,-DM**

AnsiTerm



- GEM Oberfläche
- ANSI X3.64 (VT100, VT102)
- Protokolle: KERMIT, XMODEM.
- Multi-Tasking fähig: Transfer von Dateien im Hintergrund.
- Programmierbar für Auto-Logons o.ä. durch Script-(Befehls-) Dateien.

*** 89,-DM.**

Programmierer gesucht!

Regent *Base 2.0*

Zwei leistungsfähige Komponenten in einem: GEM und SQL. Mit der professionellen Datenbank Sprache SQL und den komfortablen GEM Formularen, steht Ihnen in Regent Base ein flexibles relationales Datenbank System zur Lösung Ihrer Probleme zur Verfügung. Zum Lieferumfang gehört REGENT WORD II**. ***349,-DM.**

The Informer

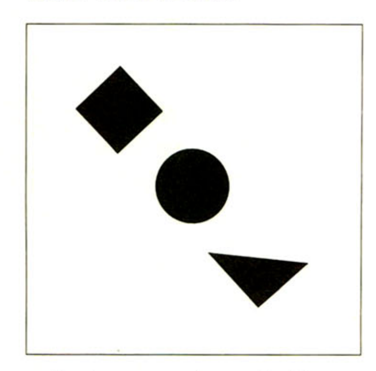
The Informer ist eine leicht zu bedienende Datenbank unter GEM. Alle Funktionen sind schnell über die Maus oder die Tastatur (wahlweise) aufzurufen und machen eine Programmierung überflüssig. The Informer ist unglaublich schnell, da alle Daten im Speicher verfügbar sind. Zum Lieferumfang gehört REGENT WORD II**. ***249,-DM.**

Hard Disk Sentry **NEU!**

überprüft Ihre Festplatte, erkennt und behebt vorhandene Fehler, BEVOR sie einen Schaden anrichten können. Eine weitere Funktion ist die Optimierung der Datei-Belegung, um schnellere Zugriffszeiten zu erreichen. ***139,-DM.**

** Regent Word II ist eine funktionelle Textverarbeitung unter GEM. Ein integrierter Taschenrechner und die Möglichkeit Serienbriefe zu erstellen, sind nur einige der vielen Funktionen.

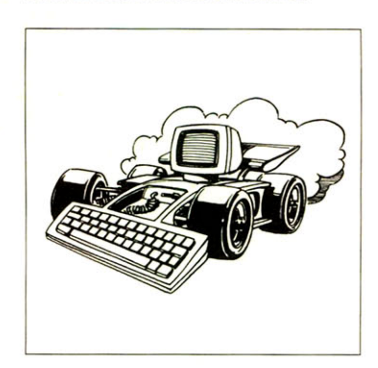
Hard Disk Toolkit



- Daten werden direkt (gepackt) auf der Diskette gespeichert.
- Verify des Backups.
- Sichern/Zurückschreiben von einem Ordner, Baum oder der gesamten Partion.
- Sichern von geänderten Dateien.

*** 89,-DM.**

Hard Disk Accelerator



- Die Hard Disk ist bis 300% schneller.
- Arbeitet schon mit 32kRAM.
- Cache Größe kann leicht verstellt werden.

*** 98,-DM.**

*Unverbindliche Preisempfehlung

COMPUTERWARE

Gerd Sender, Moselstraße 39, 5000 Köln 50, Tel.: 0221-392583. **Schweiz:** Senn Computer AG, Langstraße 31, CH-8021 Zürich

Für Ästheten: Der Painters-Algorithm

Besonders elegant kann man den Ansatz des z-Buffer-Algorithmus eigentlich nicht nennen, denn jedes Pixel muß auf seine Tiefe geprüft werden, alle Pixel, die überhaupt vorkommen, müssen mindestens einmal bearbeitet werden. Wenn nur zwei Polygone gezeichnet werden sollen, die dazu noch weit auseinanderliegen, müssen trotzdem alle Pixel, die in diesen Polygonen liegen, einzeln getestet werden, obwohl ganz offensichtlich ist, das diese Objekte sich nicht verdecken können - Kurz: der z-Buffer-Algorithmus ist zwar relativ effizient, aber irgendwo eine 'Brute Force'-Methode, also rohe Gewalt. Der Maleralgorithmus, wie die deutsche Übersetzung seines Namens lautet, arbeitet da ganz anders. Er nutzt, wenn auch nur in bescheidenem Maße im Vergleich zu wiederum komplexeren Algorithmen, die Eigenschaften der Szene, die zu zeichnen ist, aus. Der Name bezeichnet das Prinzip: Ein Maler malt auf die Leinwand zuerst die am weitesten hinten liegenden Teile der Szene und übermalt sie dann teilweise mit weiter vorne liegenden Bildteilen. Genauso tut es der Painters-Algorithm:

Stelle fest, wie weit hinten jedes Objekt liegt, sortiere und male die Objekte dann von hinten nach vorne.

Das Problem ist klar: Wie stellt man fest, wie weit hinten ein Objekt liegt? Schließlich hat ein Haus ja nicht nur eine Tiefe, sondern es beginnt z.B. irgendwo vorne im Bild und hört im Hintergrund auf... Das ist gar nicht so einfach. Aber dazu später mehr.

Der Hauptunterschied zum z-Buffer-Verfahren liegt darin, daß die Teile des Bildes, die Objekte, bzw. die Polygone, aus denen die Objekte bestehen, nicht in Pixel zerlegt, sondern als Ganzes behandelt oder, nötigenfalls, in mehrere Polygone zerteilt werden. Immer wird eine Fläche aber als Ganzes behandelt, das Ergebnis des Algorithmus ist nicht ein aus Pixeln bestehendes Bild, sondern eine eindeutig nach z-Koordinaten sortierte Liste von Polygonen, die dann einfach nur noch in der sortierten Reihenfolge gezeichnet werden müssen; der Unterschied läßt sich auch an der Komplexität der verwendeten Zeichenbefehle ablesen: Für den z-Buffer-Algorithmus braucht man nur einen Befehl zum Setzen von Pixeln, da das Bild, egal wie kompliziert es ist, immer in Pixel zerlegt wird. Das Ergebnis des Maler-Verfahrens kann mit komplexen Befehlen wie 'Zeichne gefülltes Polygon' dargestellt werden. Der Painters-Algorithm profitiert also zum Beispiel von schnellen Grafikprozessoren, die automatisch und ohne Prozessoraktivität gefüllte Polygone zeichnen können, der z-Buffer-Algorithmus kann solche Hardware-Grafikerweiterungen nicht nutzen.

Im allgemeinen ist der Maler-Algorithmus rechenaufwendiger, was durch schnelle Grafikhardware zum Teil ausgeglichen werden kann. Dafür ist der Speicherbedarf gering, die Effizienz immer noch gut. Im Verhältnis zu anderen Verfahren ist auch die Komplexität des Algorithmus erträglich, die Erweiterbarkeit jedoch auch nicht besser (eher schlechter) als beim z-Buffer-Algorithmus. Schatten oder Transparenz sind bei beiden gleich schwer zu implementieren. Dennoch, auf Mikrocomputern ist dieses Verfahren mit Abstand das beliebteste, auch wenn sich das mit steigenden Speichergrößen in Zukunft sicherlich ändern dürfte. Vor allem kann man den Maler-Algorithmus auch in unvollständigen Varianten implementieren, die einfache Objekte schnell und korrekt malen, bestimmte schwierige Probleme aber nicht lösen können. In Bild 6 sehen Sie die typischen Schwierigkeiten für Hidden-Surface-Algorithmen: Sich durchstoßende oder gar zyklisch überlappende Objekte. So etwas mag eine einfache Implementierung des Painters-Algorithm gar nicht, genausowenig konkave Objekte. Man kann die Probleme zwar lösen, doch immer auf Kosten der Einfachheit und vor allem der Effizienz. Zur Erinnerung: Da der z-Buffer keinerlei Eigenschaften der Szene nutzt, sondern sie in Einzelpixel auflöst, gibt es derlei Probleme dort nicht. Einfache Entscheidungshilfe: Haben Sie genug Speicher für einen z-Buffer, dann nehmen Sie einen, wenn nicht, versuchen Sie's mit einem Maler-Algo-

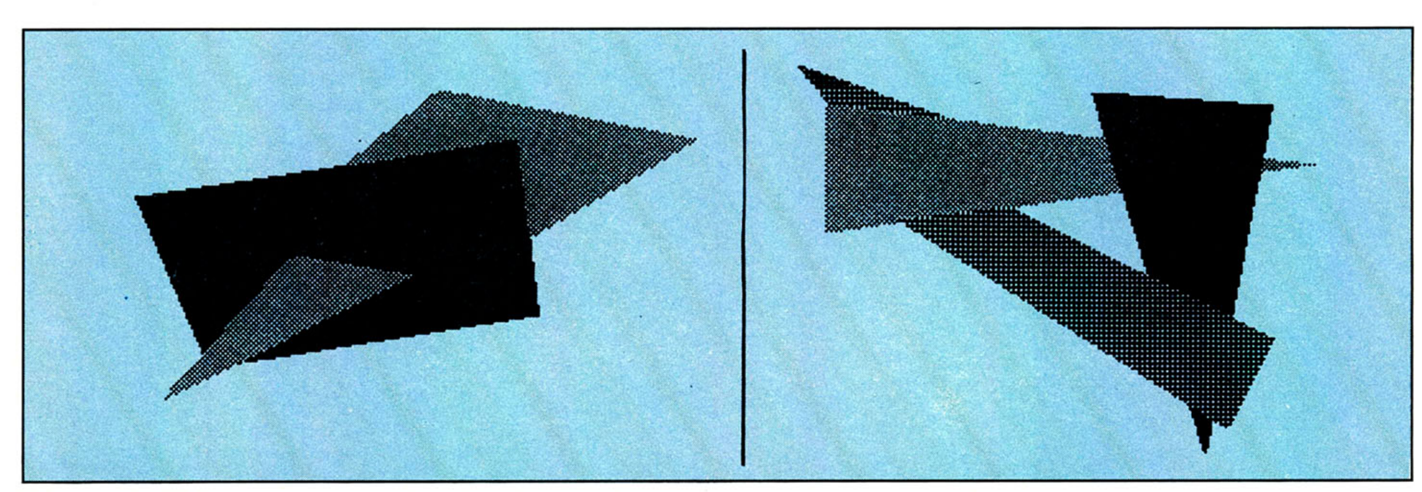


Bild 6: Testfälle für Hidden-Surface-Algorithmen. Sich durchstoßende oder zyklisch überlappende Polygone sind für viele Verfahren ein unlösbares Problem. In der Praxis kommen solche Fälle je nach Anwendung durchaus öfter vor.

Beispiel für dem Bildaufbau des "Malers"

rithmus, besonders, wenn Sie einen schnellen Grafikprozessor haben... CAD 3D zum Beispiel nutzt aus Effizienzgründen ebenfalls einen unvollständigen 'Maler', schließlich hat ja nicht jeder ST über 1 MByte Speicher.

Zuerst einmal soll der Ablauf einer vollständigen 'guten' Implementierung dargestellt werden, später werde ich noch verraten, welche Schritte Sie in einfachen, 'fehlerhaften' Varianten weglassen können.

Zuerst einmal werden alle Polygone, so wie sie sind, nach der kleinsten vorkommenden z-Koordinate sortiert. Dazu eignet sich zum Beispiel sehr gut ein Quicksort-Algorithmus. Wenn Sie Glück haben, können Sie die Polygone jetzt schon zeichnen (mit den am weitesten hinten liegenden beginnend natürlich) und alles stimmt. Leider wird das nur bei den allerprimitivsten Szenen, in denen es keinerlei Polygone gibt, die sich irgendwie überlappen, so funktionieren. Also weiter. Sie fangen hinten in der Liste an. Testen Sie das letzte Polygon in einem z-Minimax-Test gegen das nächste (d.h. korrekterweise gegen alle vor dem aktuellen Polygon liegenden Polygone). Halt, Stop. Z-Minimax-Test ? Ein Minimax-Test vergleicht die größten und kleinsten Koordinaten - zum Beispiel zweier Boxen - miteinander und stellt fest, ob sie sich überschneiden. In diesem speziellen Fall interessiert uns, ob sich die Boxen um die Polygone in der z-Koordinate überschneiden, ob also das letzte Polygon in der Liste auch wirklich ganz und vollständig hinter dem nächstfolgenden liegt. Da wir ja nur nach der kleinsten z-Koordinate sortiert haben, könnte es sein, daß die größte z-Koordinate des letzten Polygons größer als die kleinste z-Koordinate des vorletzten ist (Bild 7b). In diesem Fall wäre es möglich, daß das letzte Polygon das vorletzte und eventuell auch andere, die noch vor dem vorletzten in der Liste liegen, verdeckt. Um dies auszuschließen, müssen weitere Tests durchgeführt werden. Nur wenn sichergestellt ist, daß das letzte Polygon der Liste kein anderes verdecken kann, darf es gezeichnet und aus der Liste entfernt werden.

Das Prinzip noch einmal kurz zusammengefaßt, bevor wir uns den weiteren Tests zuwenden: Zuerst werden die Polygone nach einem einfachen Kriterium sortiert, das keinesfalls sicherstellt, daß sich die Polygone nicht mehr gegenseitig überlappen. Die entstehende Liste wird von hinten nach vorne abgearbeitet, ein Polygon nach dem anderen gezeichnet. Vor dem Zeichnen muß aber sichergestellt sein, daß das hinterste Polygon kein anderes überdeckt, da ja alle anderen wie auf einer Leinwand über die bereits gezeichneten gemalt werden und diese so ganz oder teilweise verdecken. Sinnvollerweise ordnet man die Tests so, daß zuerst die wenig rechenaufwendigen Versuche zur Bestätigung der richtigen Reihenfolge gemacht werden. Ist ein Beweis erfolgreich, kann das Polygon gezeichnet werden, und man kann sich

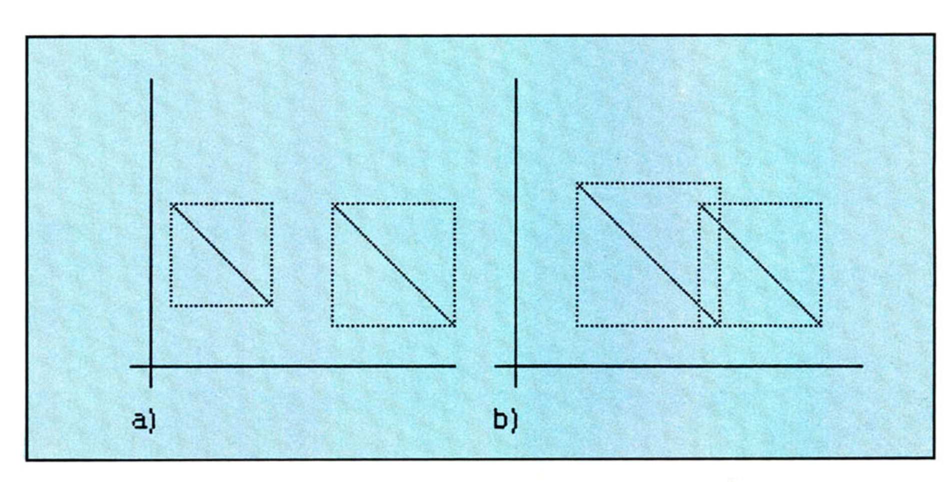


Bild 7: Zum Minimax-Test. In Fall a) ist es unmöglich, daß die Polygone sich Überlappen, da sich die umgebenden Boxen nicht berühren. In Fall b) ist eine Berührung möglich, muß aber durch weitere Tests bestätigt werden, da so ja nur eine Dimension geprüft wurde.

den Rechenaufwand für die schwierigen Tests sparen. Der allereinfachste Test ist, wie bereits ausgeführt, der Minimax-Test, da hier nur verglichen werden muß, ob das Polygon vollständig hinter den anderen Polygonen liegt oder nur teilweise.

Wenn ein Polygon nur teilweise hinter den anderen liegt, sollte man als nächstes einen Minimaxtest in x- und y-Richtung durchführen. Wenn hierbei keine Überlappung zwischen dem aktuellen und einem weiter vorne liegenden Polygon festgestellt werden kann, kann das hintenliegende auch kein anderes verdecken. Wie auch ? Von vorne gesehen, ist es in diesem Fall ja vollständig sichtbar. Wenn Sie z.B ein Polygon in der rechten oberen Ecke haben und einen Haufen in der linken unteren, kann das Polygon rechts oben die Polygone links unten beim besten Willen nicht verdecken und kann somit gezeichnet werden. Der z-Buffer müßte ein solches Polygon trotzdem in Pixel auflösen (wie bereits oben erwähnt). Beachten Sie bitte, daß ein erfolgreicher Minimax-Test in zwei Dimensionen nicht bedeutet, daß sich die Polygone selbst schneiden müssen. Es werden ja nur zwei Boxen um die Polygone herum verglichen. Dadurch ist nur der negative Ausgang des Tests beweiskräftig: Wenn sich nicht einmal die Boxen um die Polygone schneiden, dann tun's die Polygone selbst natürlich auch nicht. Vergessen Sie nicht, daß Sie auch diese Tests mit allen vor dem aktuellen Polygon liegenden Polygonen durchführen müssen.

Der nächste Test ist erheblich komplizierter. Hier müssen Sie nämlich feststellen, ob sich die Polygone schneiden, oder, etwas exakter formuliert, ob ein Polygon vollständig auf einer Seite der durch das andere Poygon gebildeten Ebene liegt oder nicht.. Wenn sich die Polygone nicht schneiden, ist bewiesen, daß das hintere Polygon das vordere nicht überdecken kann. Was bedeutet, daß Sie das hintere Polygon getrost zeichnen und vergessen können. Wie geht's ? Am besten holen Sie tief Luft und die Zeitung mit der Folge über Clipping hervor. Das Problem ist nämlich nichts anderes als ein Clipping-Test der beiden Polygone unter leicht veränderten Bedingungen; es gibt nur eine Clippingebene, nämlich die Ebene des einen Polygons, allerdings muß die entscheidende Begrenzung des Clipping-Windows auf dieser Ebene nicht unbedingt rechteckig sein, sondern hängt von der Polygon-Form ab. Es ist aber wahrlich nicht schwer. Sie können, alternativ, aber auch einfach die beiden Polygonebenen gegeneinander schneiden und die Schnittgerade vergleichen; das ist auch nicht schwieriger. Wenn sich die Polygone nicht schneiden, haben Sie wieder einmal einen Beweis, daß Sie das aktuelle Polygon gefahrlos zeichnen können.

Der letzte Test ist am unangenehmsten: Wenn alles andere keinen Beweis liefert, daß sich nichts verdecken kann, muß man in den sauren Apfel beißen und die Projektionen der Polygone auf die Bildebene berechnen. Dort muß man dann feststellen, ob sich die Polygone irgendwo überlappen. Wohlgemerkt: Die Polygone, nicht die umgebenden Boxen. Das ist zwar absolut nicht schwierig, aber sehr, sehr mühsam (Projizieren können wir ja; bzw., wenn die Routine, was sinnvoll ist, erst in Bildschirmkoordinaten angewandt wird, haben wir ja schon längst projiziert). Aber was hilft's?.. Wenn die Projektionen sich nicht schneiden, kann das aktuelle Polygon gezeichnet werden. Wenn keiner der Tests den nötigen Beweis liefert, wird sich Ihr Computer ärgern, denn dann müssen das

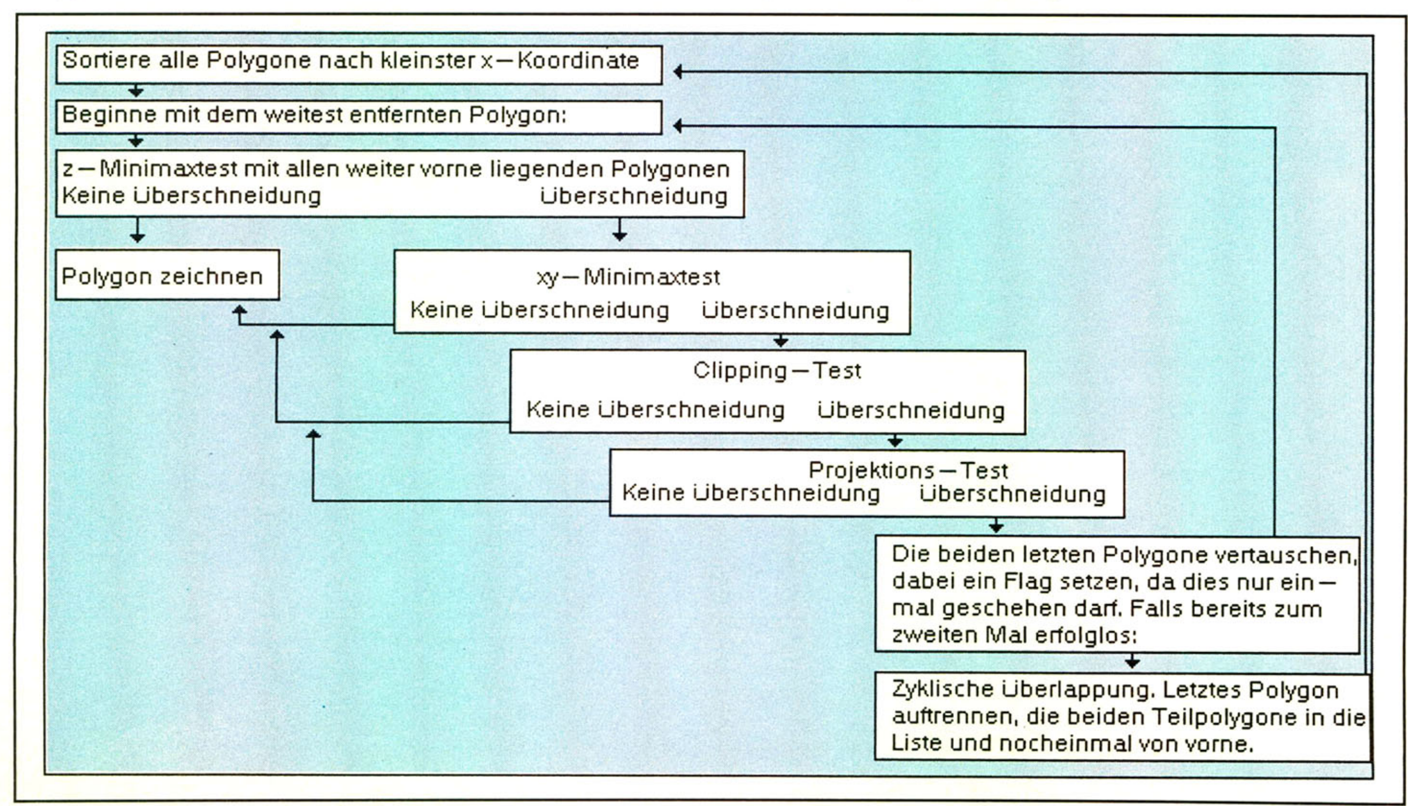


Bild 8: Diagramm des Maler-Algorithmus. Mähere Erklärungen zu den einzelnen Tests finden Sie im Text.

letzte und das vorletzte Polygon vertauscht und alle Tests wiederholt werden. Falls das immer noch nicht zu einem durchschlagenden Erfolg führt, sitzen wir in der Tinte (königsblau). Dann handelt es sich bei unserem Problem nämlich um eine zyklische Überlappung von Polygonen. In diesem Fall bleibt uns nichts anderes übrig, als das hinterste Polygon in zwei Teile zu teilen, die Liste neu zu sortieren und von vorne zu beginnen. Das Teilen eines Polygons erledigt man am besten über die Schnittgerade zweier Ebenen - Sie müssen das aktuelle Polygon entlang der Ebene des vorletzten Polygons zerschneiden.

Sie sehen, wenn Sie einen wirklich korrekten 'Maler' basteln wollen, sind Sie ganz schön beschäftigt, einfach ist das nicht mehr. Sie können, wenn Sie Bildfehler bei komplexen Objekten in Kauf nehmen wollen, natürlich einfach die schwierigeren Tests ignorieren. Viele Programme machen das so, allerdings sieht man das auch oft. In Bild 8 sehen Sie den Algorithmus in einem Diagramm.

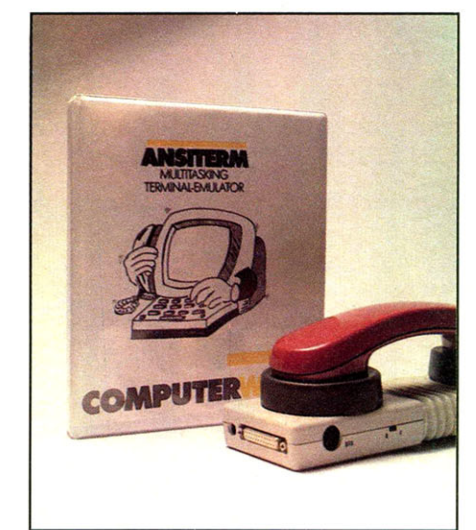
Verdeckte Oberflächen sind wirklich ein trockenes Problem. Aber da hilft leider nichts, wenigstens grundsätzlich sollte man verstanden haben, wie sowas geht, jedenfalls dann, wenn in der nächsten Folge die Erleuchtung über Sie kommen soll. Dort werden wir uns nämlich mit dem Licht auseinandersetzen, das für all diesen Ärger verantwortlich ist (Wie beim Vogel Strauß - Was man nicht sieht, ist auch nicht da). Langsam aber sicher haben wir dann die Regionen erreicht, in denen es spannend wird - die Methoden, mit denen man realistische Computerbilder erzeugen kann.

CS

ENDE

ANSITERM

DFÜ und Multitasking



ANSITERM ist, wie der Name schon sagt, ein Emulator, der ein ANSI X3.64-Terminal emuliert, wohl besser bekannt als VT 100.

Es werden auch die VT 102- und die VT 52-Emulation unterstützt. Die VT xxx- Terminals sind ursprünglich von der Firma DEC entwickelt worden, die Definitionen wurden später vom amerikanischen Normenbüro festgeschrieben.
Die Emulation ist ziemlich vollständig mit Unterstreichen, Fettschrift, blinkend und invers. Auch ein Graphikzeichensatz und ein britischer (UK) Zeichensatz sind enthalten.
ANSITERM unterstützt das Kermit- und das XMODEM-Protokoll sowie das COMPUSERVE-B-Protokoll mit Quick-B-Erweiterung. Die beiden letzteren sind wichtige Protokolle des größten amerikanischen Betreibers einer öffentlichen Datenbank. Übrigens lief der Test sehr erfolgreich.
Es sind alle gängigen Baudraten von 50 bis 19200 Baud möglich, und bei einer Rechner-Rechner-Kopplung gab es auch bei den hohen Baudraten keine Probleme.
Am unteren Bildschirmrand werden zwei Zeiten gezeigt, die Systemzeit und die Verbindungszeit, die rückstellbar ist.
Ansonsten bietet ANSITERM nicht viel Komfort, es ist kein Mailbox- sondern ein TERMINAL-Programm. Beim Programmstart präsentiert sich ANSITERM scheinbar als TOS-Programm. Erst wenn man den linken Mausknopf betätigt, erscheint eine Menüleiste, über die die Funktionen des Programms erreichbar sind. Dazu gehören neben den Parametern für Schnittstelle und Bildschirm die Möglichkeiten, ein Druckerprotokoll einzuschalten, die Verbindungszeit zurückzustellen, die Zeit für eine Bildschirmschonschaltung festzulegen und anderes. Leider ist nicht vorgesehen, die empfangenen Daten auf Diskette zu sichern, und es gibt auch keine Möglichkeit, die im normalen Terminalmodus empfangenen Daten später noch zu edieren bzw. abzuspeichern, da ein integrierter Speicher und Editor fehlen.
Dieses Manko wird nur teilweise durch die Möglichkeit wieder wettgemacht, eine Shell bzw. z.B. einen Editor zu definieren, den man von ANSITERM aufrufen kann. Über Alt-Esc kann man während einer Protokoll-Übertragung in dieses definierte Programm springen. Möglich wird dies durch ein mitgeliefertes Programm namens RTXBOOT. PRG. Es handelt sich hierbei um einen Multitasking-Kernel, der am besten in einem Auto-Ordner installiert wird. Unter anderem fängt dieses Programm auch so manche GEM-Exception ab, in ATARIanisch "Bömbchenfehler" genannt.
Ansonsten sorgt es dafür, daß man von ANSITERM aus andere Programme aufrufen kann.
Ähnlich wie andere DFÜ-Programme bietet ANSITERM die Möglichkeit, die Funktionstasten doppelt zu belegen und sogenannte Skriptdateien auszuführen. Diese Definitionen müssen allerdings mit einem externen Editor erstellt und in speziellen Dateien abgelegt werden, da ANSI-TERM wie bereits erwähnt über keinen eingebauten Editor

File Options
Load parameters...
Save parameters...
Load Macros...
Execute Script...
File Capture...
File Send...
DC2/DC4 Protocol
XMODEM receive...
XMODEM send...
Kermit GET...
Kermit SEND...
Kermit RECEIVE
Kermit FINISH
Kermit Image Mode
Quit

Bild 1: Die File-Options von ANSITERM

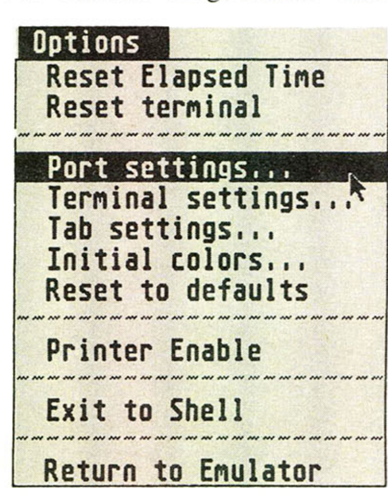


Bild 2: Das Options-Menü von ANSITERM

verfügt. Mit Hilfe der Skriptdateien kann man zum Beispiel automatische Einlogprozeduren schreiben, oder sogar einen ganzen Hostzugriff automatisieren. ANSITERM kennt zu diesem Zweck 25 verschiedene Kommandos, mit deren Hilfe man alles erledigen kann, vom kompletten Filetransfer bis zum Weckton für den User, der sich während einer halbstündigen Übertragung süßen Träumen hingibt.

ANSITERM wird mit einem deutschen Handbuch geliefert, über dessen Qualität man geteilter Meinung sein kann. Einerseits werden detaillierte Informationen über die verschiedenen verwendeten Protokolle und Standards gegeben, andererseits werden so wichtige Dinge wie das RTXBOOT.PRG mit keiner Silbe erwähnt. Dafür enthält es aber sowohl ein Inhaltverzeichnis, ein Stichwortverzeichnis sowie einige nützliche Literaturhinweise und ein gutes Begriffslexikon.

Neben dem eigentlichen Terminalprogramm, dem Multitaskingkernel und einer Parameterdatei befinden sich auf der gelieferten Diskette noch je ein Beispiel für Macros und Scriptfiles sowie noch die englisch-sprachige Version des Handbuchs.

Da ANSITERM nicht kopiergeschützt ist, läßt sich eine Sicherheitskopie anfertigen und auf einer Harddisk unterbringen.

ANSITERM ist ein Produkt von Beckemeyer Tools, einer US-Firma, deren Produkte fast alle mit Multitasking zu tun haben. Schon seit über einem Jahr existiert die RTX-Shell, die auch schon seit einiger Zeit durch die MT C-Shell ergänzt wird. Zusammen bieten die beiden praktisch UNIX auf dem ST, nehmen allerdings zusammen auch fast 800K in Anspruch. Mit dem VSH-Manager, einem weiteren Produkt von Beckemeyer, ist es möglich, über ein Accessory mehrere Hintergrundtasks auszuführen, während man vorne mit einem GEM-Programm arbeitet, eine besonders für Entwickler interessante Sache.

Zum guten Schluß bleibt anzumerken, daß ANSITERM seine Tests bestanden hat, die Protokollübertragungen einwandfrei funktionieren und es ein zuverlässiges Werkzeug darstellt, auch wenn man oft den Komfort anderer Terminalprogramme vermißt. Einzigartig ist die Mög-lichkeit, während einer Protokollübertragung mit einem anderen Programm zu arbeiten, allerdings gibt es hier einige Einschränkungen. So liefen nur TOS-Anwendungen zuverlässig, GEM-Programme sind wohl (hoffentlich noch) nicht vorgesehen. Für Leute, die oft große Datenmengen übertragen müssen und die die Zeit nicht verträumen wollen oder können, ist ANSITERM ein guter Kandidat fürs Softwareregal.

Christian Schmitz-Moormann

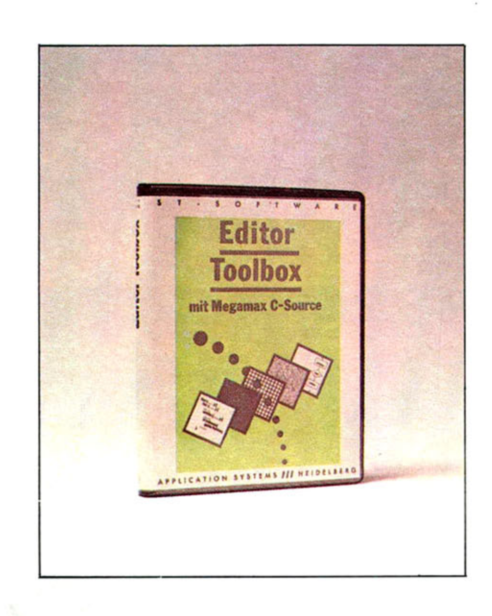
Terminal Type	ANSI	VT-52
Auto Repeat	NO	YES
Auto Wrap	NO	YES
Display CTRL Chars	NO	YES
Clear screen on FF	NO	YES
Tabs to Spaces	NO	YES
Keypad	Hex	Normal
File Capture/Send	7-bit	8-bit
Strip LF for upload	NO	YES
BS is destructive	NO	YES
Cursor	UNDERLINE	BLOCK
	FLASHING	STEADY
	OFF	ON

CRT Saver Time: 30_
Shell:csh_______________________
OK CANCEL

Bild 3: ANSITERM besitzt vielfältige Einstellungsmöglichkeiten

ENDE

EDITOR TOOLBOX



Was in der IBM-Welt längst ein alter Hut ist, erlebt jetzt auf dem ST eine Renaissance. Das Kind heißt Editor Toolbox und kommt von Application Systems /// Heidelberg. Für 149 DM bekommt man einen Editor und dessen Quellcode in Megamax C.

Wer bisher neidisch zu den Turbo-Pascal-Programmierern auf dem PC herübergeschielt und Gefallen an dem Toolbox-Konzept gefunden hat, wird jetzt auch auf dem ST fündig. Die Firma Application Systems, die den Megamax C-Compiler in Deutschland vertreibt, bietet inzwischen einige Erweiterungen und Programme an, bei denen auch der Quellcode in C mitgeliefert wird. Dies ist für alle Anwender und Freaks, die selbst in C programmieren können und im Besitz des Megamax Compilers sind, ein interessantes Feature. Um es vorwegzunehmen: Bedenkt man den Preisunterschied von 85 DM zu 149 DM zwischen Tempus und der Editor Toolbox, dann muß man eigentlich Verwendung für den Quellcode haben, sonst ist man mit Tempus sicher besser bedient.

Was bekommt der Anwender geboten ?

Einen Bildschirmeditor, der sich durch seine Funktionalität von vielen anderen Produkten dieser Art abhebt. Ich meine damit, daß nicht nur die normalen Editorfunktionen wie Einfügen, Löschen, Blöcke verschieben usw. geboten werden, sondern zusätzlich noch ein Terminalprogramm enthalten ist, mit dem man seine Texte direkt über einen Akkustikkoppler oder ein Modem an eine Mailbox schicken kann.

Alle Funktionen des Editors werden auf den 32 Seiten der deutschen Bedienungsanleitung ausführlich erläutert. Wer schon mit verschiedenen Texteditoren gearbeitet hat, braucht das Handbuch nur in seltenen Fällen zu Rate zu ziehen. Der Editor verwendet eine Menüleiste, von der aus alle Funktionen angesprochen werden können. Zusätzlich kann man aber auch alle Kommandos über diverse Tastenkombinationen erreichen. Dabei hat es mir sehr gut gefallen, daß in den Menüs immer die Tastenkombination, mit der eine Funktion ausgelöst werden kann, angezeigt wird (siehe Bild). Daran könnte sich Tempus ein Beispiel nehmen. Wo wir gerade bei Tempus sind: In meinen Augen ist Tempus im Augenblick der hellste Stern am Editorhimmel. Deshalb mußte er auch in diesem Test als Vergleich herhalten. Ich gehe davon aus, daß die Leser schon von Tempus gehört haben; wenn nicht: siehe ST-Computer März 1987, Seite 70, Testbericht über Tempus (Hallo Stefan). Ein wesentliches Merkmal von Tempus möchte ich an dieser Stelle noch erwähnen: die überdurchschnittliche Verarbeitungsgeschwindigkeit. Speziell in diesem Punkt habe ich die beiden Programme miteinander verglichen und zwar anhand des Quelltextes der Editor Toolbox, der stolze 5712 Zeilen lang ist. Die ermittelten Zeiten entnehmen Sie bitte der Tabelle.

Ein paar Worte möchte ich noch über die Funktionalität verlieren. Es wird alles geboten, was man von einem Texteditor zum Bearbeiten von Programmtexten erwartet. Man kann suchen und ersetzen, Blöcke definieren und diese kopieren, löschen und verschieben, vier Marken setzen und dann direkt zu den Marken springen, zwischen 2 Zeichensätzen (8*16 und 8*8) wählen, die Tabulatorweite setzen usw. Erwähnenswert ist noch, daß man einen Block einrücken kann. Diese Funktion ist besonders für die Strukturierung von Programmtexten nützlich. Leider besteht keine Möglichkeit, den Namen einer Textdatei in der Kommandozeile zu übergeben. Dieser Mangel macht die Verwendung des Editors zusammen mit der Megamax-Shell unhandlich. Man muß immer zuerst den Text laden und kann nicht sofort loslegen, wie man es vom Megamax-Editor oder von Tempus kennt. Aber es gibt ja den Quellcode...

geht weiter...



In diesem braucht man jedoch nicht herumzubasteln, um Abstürze zu beseitigen, zumindest gab es im Test keine Bomben zu sehen.

Was bekommt der Programmierer geboten ?

Wie man sich vorstellen kann, ist ein kompletter Editor mit Terminalemulation nicht gerade ein Mickey Mouse-Programm. So wird es den einen oder anderen überraschen, daß für die 40 KByte Programm 240 KByte Quelltext nötig sind. Dazu muß man sagen, daß der Quelltext außer-gewöhnlich umfangreich dokumen-tiert ist. Hinter fast jeder Zeile steht ein Kommentar, und vor den einzelnen Funktionen werden jeweils die Parameter und die Wirkungs-weise erklärt. Diese Tatsache erleichtert das Verständnis der einzelnen Programmteile ungemein. Leider ist dies aber auch die einzige Unter-stützung, die geboten wird, um den Code zu verstehen. Das Design, das dem Programm zugrunde liegt, das Zusammenspiel der einzelnen Mo-dule, welche Datenstrukturen benutzt werden und alles andere, was noch zum Verständnis des Quelltextes bei-tragen könnte, fehlt. Es bleibt nichts anderes übrig, als mühsam den Quelltext zu durchforsten um nach und nach die Zusammenhänge zu verstehen.

Was mich am meisten stört, ist die Gestalt des Quelltextes. Es sind - wie schon erwähnt - ca. 240 KByte, die jedoch in einer einzigen Datei stehen. Selbst Standarddefinitionen, die jeder vernünftige Entwickler mit '#include' in seinen Text einbindet, stehen in dem Sourcefile 'TOOLBOX.C'. Diese Art zu programmieren erinnert mich an meine ersten Gehversuche in Sachen Computer, auf einem VC-20 in BASIC. Allerdings hatte ich damals keine andere Wahl, als alles in einem Stück in den Speicher zu hacken. Warum der Autor der Editor Toolbox, Martin Backschat, seinen Quelltext nicht in mehrere kleine Objektmodule unterteilt hat, ist mir schleierhaft. Gerade Megamax C unterstützt doch hervorragend die Modularisierung. Die Make-Funktion kompiliert nur die edierten Module, und bei einer kleinen Änderung muß man nicht 240 KByte neu übersetzen, sondern vielleicht 10 KByte, und spart jede Menge Zeit. Auch wenn man den Text ediert, ist es angenehmer, in kleineren Stücken zu arbeiten, als jedesmal die komplette Source, die, ich wiederhole es nochmal, 240 Kbyte lang ist, in den Editor zu laden. Der Code selbst ist trotz der vielen Kommentare für einen Anfänger nicht leicht zu verstehen. Es fehlt halt eine Programm-Dokumentation. Leider sind auch noch ca. 200 Zeilen Assembler enthalten, die zwar die Verarbeitungsgeschwindigkeit, nicht jedoch die Lesbarkeit des Programms erhöhen. Ich will nicht am Programmierstil von Martin Backschat mäkeln, aber ich habe schon übersichtlichere Programme gesehen. Es gibt einige Stellen im Code, an denen man Verbesserungen vornehmen kann und sei es nur um die Optik zu verbessern. Mit Makros wurde richtig gegeizt. Schade, warum eigentlich ? Genug gemeckert, sicher gibt es auch an den Programmen, die ich in der ST-Computer veröffentlicht (verbrochen ?) habe, einiges auszusetzen. Egal, wie fit man in einer Programmiersprache ist, es gibt fast immer einen, der besser drauf ist.

Alle Befehle lassen sich über Tastaturkombinationen aufrufen

Programm / Aktion	Editor Toolbox	Tempus
Text laden	11 s	4 s
Text von Anfang bis Ende scrollen	343 s	221 s
Seitenweise blättern	32 s	17 s
Alle { durch _L_ ersetzen	34 s	< 3 s

Tabelle: Die Editor Toolbox im Vergleich zu Tempus

Fazit

Die Editor Toolbox stellt mehr Editor als Toolbox dar. Man kann sich zwar einige Tricks für seine eigenen Programme abgucken; um richtig durchzublicken und Modifikationen zu machen, muß man aber Zeit haben. Ich wünsche mir im wesentichen zwei Dinge: Modularisierung des Sourcecodes und eine ausführliche Beschreibung der Module und Datenstrukturen. Ansonsten ist das Paket recht gelungen. Dafür, daß fast ausschließlich in C programmiert wurde, kann man sich hinsichtlich der Geschwindigkeit nicht beklagen. Der Vergleich mit Tempus, der ja in Assembler geschrieben wurde, ist fast schon ungerecht.

JL

DER UNTERSCHIED



G RAMDISK II +

In der neuesten Version mit allen Leistungsmerkmalen, die man sich für eine Ramdisk überhaupt wünschen kann: Resetfest bis 4 MB, dabei abschaltbar; integrierter Druckerspooler, Bootcopy; Parametersave usw. Die G RAMDISK II ist im harten Einsatz bereits tausendfach bewährt.

DM 49,—



INTERPRINT II

Die universelle Anpassung für jeden Drucker hat einen integrierten Treiber, der auch alle Umlaute und Sonderzeichen nach Ihren Wünschen konvertiert. Dabei können mit Hilfe des Editors maßstabsgerechte Hardcopies auf allen Matrixdruckern und HP-Lasern erstellt werden. Auch ein Druckerspooler und unsere G RAMDISK II + sind integriert. Die Parameter sind speicherbar, und der komfortable Editor ermöglicht schnellste Anpassung.

DM 99,—

ohne Ramdisk **DM 49,—**



DISK HELP

Repariert Disketten und erlaubt in 99% der Fälle eine Wiederbelebung des gesamten Datenbestandes. Das Programm gehört neben jeden ST. Lassen Sie Ihre wichtigsten Daten nicht einfach von Laufwerken zerstören. DISK HELP ist einfach zu bedienen, korrigiert Lesefehler und rekonstruiert Files. Physikalische Fehler (Risse, Kaffee) können nicht behoben werden.

DM 79,—



Harddisk Help & extension

Wir wollen keinesfalls die Ängste um Ihre täglichen Daten schüren, aber sind Ihre Daten wirklich sicher? Wie schnell führt ein Headcrash zum Ende! Sie sollten auf ein bewährtes Backupsystem nicht länger verzichten. Auch sehr große Files können einzeln gesichert werden. Partition Backup, Tree Funktion, selected Backup und Diskoptimizer erhöhen die Effizienz dieses Programms. Auch bei der Geschwindigkeit wurden keine Kompromisse geschlossen.

DM 129,—



G COPY

Für Vielkopierer, Sicherheitskopierer, Raubkopierer und Umkopierer. Das universelle Kopierprogramm, das einfach mehr kann: G COPY kopiert alle ST-Programme und jeden Kopierschutz. G COPY formatiert alle Diskettenformate (80-87 Tracs, 9-11 Sektoren, 913 KB Disketten, Schnelladedisketten) in Hochgeschwindigkeit. Dabei erreicht G COPY eine unglaubliche Kopier- und Formatiergeschwindigkeit.

Jetzt neu: Mit Autocrack, Copy Protection, Detector und Copy Protection Emulator.

DM 99,—



FAST SPEEDER

Für alle, die ihrem Massenspeicher keine Pause gönnen wollen und lange Wartezeiten satt haben. Bringen Sie Ihre Festplatte und Diskettenstation auf Trab! Im Extremfall erreichen Sie dabei Ramdiskgeschwindigkeit! Festplatten werden bis zu 200% beschleunigt, Diskettenlaufwerke bis zu 900%. FAST SPEEDER ist einfach zu handhaben. Der optimierte Spezialalgorithmus mit Cache Prinzip arbeitet für Sie unmerklich im Hintergrund.

DM 129,—



G DISKMON II

Zum Kontrollieren, Ausprobieren, Umkopieren, Raubkopieren, Manipulieren, Interessieren und Reparieren. Ein nützliches und wichtiges Tool, damit die Diskette kein Geheimnis bleibt. Unbeschränktes Lesen, Schreiben, Editieren, Kopieren, Suchen, Ersetzen, Formatieren usw.; Schnelldruck; Bootsektorscannung in Klartextdarstellung; interaktive Hex, Dez oder ASCII Bedienung.

DM 99,—



G DATEI

Maskengenerator, einfache, effektive Benutzerführung, Help Funktionen, Rechenfelder, Druckeranpassung, Listen- und Druckmaskengeneratoren sind für G DATEI selbstverständlich. Darüberhinaus sind in G DATEI ein Taschenrechner, eine kleine Textverarbeitung für Serienbriefe und ein Expertensystem integriert. Alles Logisch aufgebaut und ohne große Kenntnisse einfach zu bedienen. Damit Sie Ihre Daten nicht nur erfassen, sondern auch bearbeiten und auswerten können, wie Sie es wollen. Nicht umsonst ist G DATEI von der Gesellschaft der unabhängigen EDV-Berater empfohlen!

DM 199,—

Handbuch (vorab - wird angerechnet) **DM 20,—**



CHESS

Das stärkste Schachprogramm für ST Computer (bei 66.000 Eröffnungszügen) bietet nicht nur Spielstärke, sondern auch Komfort. Nachladbare und erweiterbare Eröffnungsbibliothek, beliebige Bedenkzeit (bei Ausnutzen Ihrer Bedenkzeit), Zugvorschlag, Seitenwechsel, Hängespiel, Problemschach, Patreindruck und Speicherung, Blitzen usw. sprechen klar für CHESS.

DM 129,—



AS SOUND SAMPLER II

Der Sound Sampler II hat nicht nur die Presse überzeugt (Keyboard 12/87: 'verglichen mit anderen Samplern, die das 10-15fache kosten ... muß man von einer guten bis sehr guten Sample Qualität sprechen.' ST 1/88: 'Der Alleskönner'), sondern auch unsere Kunden. Und die Leistungsmerkmale sprechen für sich: Samples aufnehmen, editieren, schneiden, verknüpfen, mischen, transponieren, Echoeffekte, einbinden in Programme, 36,4 KHz Samplingrate überzeugen klar.

Software **DM 149,—**

Hardware **DM 249,—**

Soundbibliothek (10 Disketten) **DM 149,—**

IST DEUTLICH



G SCANNER

Problemloses Digitalisieren von s/w-Bildern mit Ihrem Drucker ermöglicht der G SCANNER. Die hochwertige Abtastdiode von HP garantiert die ausgezeichnete Qualität von 200 dpi. Sie wird einfach auf dem Druckknopf Ihres Druckers befestigt. Mitgeliefert wird Software, die DEGAS und Normalbilder erstellt und Bildeditieren ermöglicht. Der G SCANNER ist voll kompatibel zu STAD und SYMPATIC PAINT (Scannoption).

DM 298,—

P6 Halterung für Diode DM 35,—



Roboterarm

Der semiprofessionelle 5achsige Roboterarm wird anschlußfertig für ST, AMIGA oder PC/AT geliefert. Für Schulungszwecke, Lehr- und Demonstrationszwecke oder einfach zum Spielen bestens geeignet. Technische Daten: Höhe ca. 54 cm, industriegelb, ca. 2,5 kg schwer, einfacher Anschluß an Drukkerport. Auch über Joysticks zu steuern.

Anschlußfertig DM 269,—
Spezialnetzteil DM 99,—
Batteriesatz DM 10,—



TV Modulatoren

Schließen Sie Ihren ST an den Farbfernseher an, und sparen Sie sich einen teuren RGB Monitor. Unsere Modulatoren bringen ein scharfes Bild mit allen Mischtönen und einen Tonausgang extern und einen über den Fernseher. Das heißt: Bestes Bild und bester Tor für alle Fernseher.

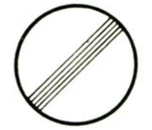
Typ A mit Monitorumschalter (ATARI-Fernseher) DM 249,—
Typ B ohne Monitorumschalter DM 239,—
Typ C FBAS Wandler für Geräte mit AV oder FBAS Eingang DM 169,—



RAM Erweiterungen

Unsere Ram Erweiterungen sind völlig neu. Sie werden ohne Lötarbeiten im Gerät gesteckt. Keine Leiterbahntrennungen - keine Treibersoftware. Die Platinen werden unbestückt geliefert und können mit 256 KBit Chips oder mit 1 MBit Chips bestückt werden. So erreicht man - je nach Wunsch - eine Erweiterung um 512 KB, 2 MB oder 4 MB. Dazu benötigen Sie nur eine Platine. Das bedeutet: zukunftssicher und problemlos.

für 520, 520+, 260, STM DM 249,—
für 1040 STF DM 249,—



LOGIC ANALYSER

Endlich ein Logik Analyser, den sich jedermann leisten kann. Dabei können wir auf die Leistungsmerkmale von 5-10fach teureren Geräten hinweisen: 16 Kanäle bis 600 KHz, komfortable GEM Software, verschiedene Triggerungsmöglichkeiten usw. Das Gerät wird auf den Romport gesteckt und anschlußfertig mit Software geliefert.

DM 448,—

NEUHEITEN



RELAS

Aktienanalyse für den privaten und professionellen Aktienanleger auf der Basis der **Relativen Stärke und Volatilität** von Aktienkursen — **Charts** wahlweise über Bildschirm und Drucker — **Mischen von Charts** einer Aktie und Branche zum Vergleich — **Kurseingabe nur einmal wöchentlich (!)** Als Basis dienen die letzten 27 Wochenschlußkurse — Umfangreiche Bereinigungsfunktion bei Kapitalveränderungen von Aktien — **Listen** der Relativen Stärke mit Mittelwert von Kursen und Analyse der Veränderung der Rangfolge der Relativen Stärke über 3 Wochen (wahlweise Bildschirm oder Drucker) — Zu jeder Aktie können **Höchst- und Tiefstkurse** sowie eigene **Notizen** eingegeben werden — **Depotverwaltung** für den privaten und semiprofessionellen Aktienanleger über ein **bedienungsfreundliches Karteikartensystem** auf dem Bildschirm mit **Gewinn- und Verlustrechnung** Brutto — **Depotübersicht** wahlweise auf Bildschirm oder Drucker aufgeschlüsselt nach Einzelgattung oder als Gesamtübersicht — Umfangreiche **Lösch- und Änderungsfunktion** für die Daten

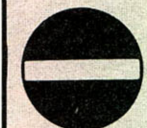
Handbuch vorab: DM 40,— DM 398,—



SYMPATIC PAINT

Der ATARI ST als Grafik Workstation der Superlative! Alle Funktionen, die für ein effektives und komfortables Zeichnen, Malen und Konstruieren erforderlich sind, wurden integriert. Der Benutzer hat nicht nur die Möglichkeit der Erstellung einer individuellen Grafiktoolbox, sondern auch eine Animationsselektion, mit der Bilder laufen lernen. Bilder können dabei mit dem G SCANNER und EASYTIZER eingelesen werden. Selbstverständlich kann die laufende Animation mit Sound unterlegt werden. Dabei können nicht nur alle ATARI Sound Register gezogen werden, sondern auch digitaler Sound vom AS SOUND SAMPLER II eingebaut werden. Trickfilme mit digitalem Sound!

Handbuch vorab: DM 30,— DM 298,—



GEM Retrace Recorder

Der Knüller! Wie von Geisterhand bewegt sich die Maus, laufen Kommandos ab, werden Aktionen gestartet. Der GEM Retrace Recorder ermöglicht es, jegliche Aktion (Mausbewegung, Menüs, Klicks, Tastatur usw.) aufzuzeichnen, und auf Tastendruck jederzeit original zu wiederholen. Das heißt: GEM Makros erstellen für Aktionen, die ständig wiederholt werden (auch in jedem Programm); super Auto Boots erstellen; selbstlaufende Demos von jedem Programm ohne Programmeingriffe und Kenntnisse u.v.m. Die Anwendungsmöglichkeiten sind fast unbegrenzt!

DM 99,—



Hardware Uhr

Die Uhr wird ohne Lötarbeiten im Rechner eingebaut, so daß der ROM Port frei bleibt. Sie haben ständig die aktuelle Zeit und das aktuelle Datum! Die Lithium Batterie garantiert hohe Ganggenauigkeit und 10 Jahre Laufzeit bei voller Schaltjahrerkennung. Der sensationelle Preis für die anschlußfertige Uhr beträgt nur

DM 79,—

G DATA

Siemensstraße 16 4630 Bochum 1
Telefonische Bestellungen:
023 25 / 6 08 97

Schweiz: Senn Computer AG
Langstr. 31
CH-8021 Zürich

Österreich: Computershop Rittner
Hauptstr. 34
A-7000 Eisenstadt

BENELUX:

G DATA BeNeLux
Postbus 70
NL-2000 AB
Haarlem
Tel. 023 / 32 13 31

COUPON

Hiermit bestelle ich ☐ Informationsmaterial (DM 2,— beiliegend)
☐ 2 Demodisketten und Infomaterial (DM 10,— beiliegend)

☐ per Nachnahme (ca. DM 6,50 Versandkosten)
☐ Verrechnungsscheck liegt bei (+ DM 5,— Versandkosten)

Absender:

NEU

EXTENDED VT52-EMULATOR

Bekanntlich können Computer und Programme gar nicht schnell genug sein, um eingefleischte Hacker zu befriedigen. Dies gilt nicht zuletzt auch für die Bildschirmausgabe, die Teil des Betriebssystems und darum nicht so einfach durch eigene Routinen zu ersetzen ist. Wie man die Textausgabe um bis zu Faktor sechs beschleunigt und dergleichen mehr, zeigt ein in Assembler geschriebenes Programm, dessen erster Teil in dieser Ausgabe der ST Computer veröffentlicht wird.

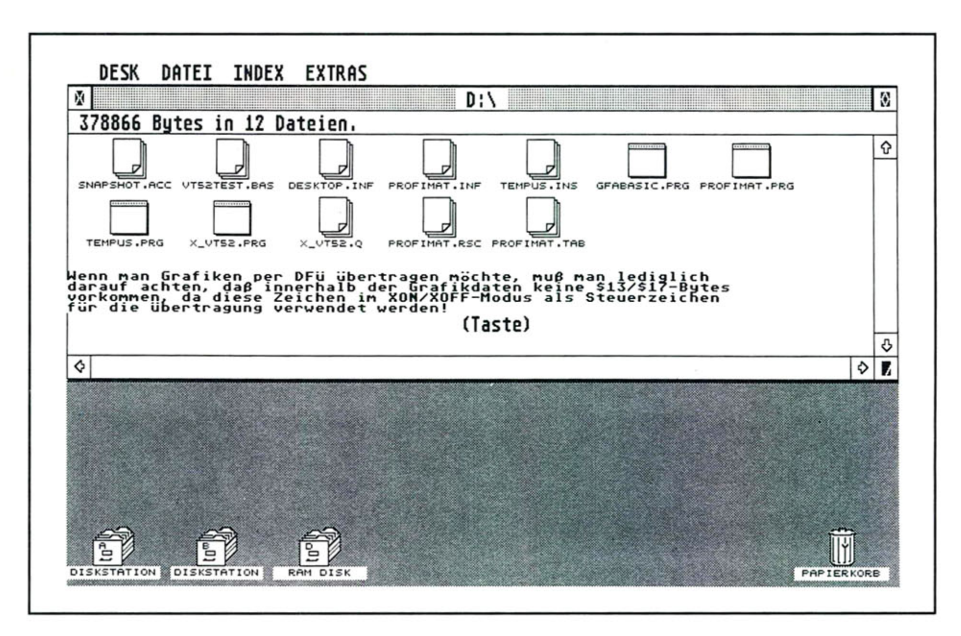


Bild 1. Auch das Anzeigen und Übertragen (DFÜ) von Bildern in DOODLE-Format ist möglich.

Mag sein, daß es etwas ungewöhnlich ist, den Anfang mit dem Ende zu verbinden (Sokrates' Lieblingsbeschäftigung einmal ausgekreist), aber da dieser Artikel ob seiner Länge nur häppchenweise publiziert werden kann, sollten Sie wenigstens jetzt schon erfahren, was Sie darin erwartet und was Sie damit anfangen können. Es handelt sich bei xVT52.PRG um einen Patch des für "normale" Textausgabe (also solche ohne Attribute) zuständigen VT52-Emulators. Das heißt, es ist eigentlich kein Patch mehr, sondern viel mehr ein komplett neuer Emulator, denn wenn er erst einmal installiert ist, kann sich der Original-Emulator auf Rente begeben, weil keine einzige Funktion des TOS ihn mehr eines Bytes würdigt. Dafür ist das komplett in Profimat-Assembler geschriebene Programm wesentlich leistungsfähiger im Funktionsumfang und auch bis zu sechsmal schneller als sein in C geschriebener Kollege. C erzeugt sicherlich kompakten und recht schnellen Code, wenn man der Perversion übersichtlicher Programmierung mächtig ist, aber schon das Runtime-Gerüst eines C-Programmes ist bei weitem größer als der komplette xVT52-Emulator, und was die Geschwindigkeit angeht, spricht ein Zeitvergleich zwischen altem und neuem Emulator für sich.

Features

Sind Sie Mitglied der immer größer werdenden DFÜ-Fan-Gemeinde? Dann haben Sie es bestimmt schonmal vermißt, daß der VT52 zwar ASCII-Texte übermitteln kann, nicht aber Grafiken. Häufig kommt es auch vor, daß man eben mal einen Text unterstreichen möchte, um ihn hervorzuheben. Vor derlei Sonderwünsche hat das TOS das VDI gesetzt; und das ist nun nicht eben ein Ausbund an Bedienungsfreundlichkeit. In aufwendigeren Tabellen kann es auch nötig werden, mehr als nur eine Cursorposition zwischenzuspeichern (mit ESC j); hatte man doch keine legale Möglichkeit im VT52, an die derzeit adressierte Spalte/Zeile heranzukommen... Ein häufig geäußerter Benutzerwunsch ist auch das pixelweise Verschieben des Bildschirmbereiches (Finescrolling genannt), das sich nun sehr einfach erreichen läßt - sofern Sie zu den achtzig Prozent der ST-Besitzer zählen, die einen SM124 ihr monochromes eigen nennen. Dagegen spielt es keine Rolle, ob nun ein 260er oder ein Mega ST4 Ihren Schreibtisch ziert: xVT52 läuft auf allen Konfigurationen und TOS-Versionen.

Im übrigen mag Ihnen diese Serie dazu dienen, das eine oder andere dazuzulernen, derweil zumindest die wichtigsten Programmteile nochmal ausführlich auseinandergenommen werden, um ihre Funktionsweise zu erläutern.

Installation

Für den Anwender reduziert sich der Aufwand zum Installieren des Programmes auf einen Maus-Doppelklick, den man sich aber auch ersparen kann, wenn man xVT52.PRG in den AUTO-Ordner des Bootlaufwerks legt. Um den Emulator im Betriebssystem zu verankern, ist allerdings erheblicher Aufwand zu betreiben: nicht weniger als alle vom TOS benutzten TRAP-Vektoren sind umzubiegen, um alle Aufrufe abzufangen, die irgendwas mit Bildschirmausgabe zu tun haben. Und das sind derer reichlich viele...

Des' Trap ich bieg', des' GEM ich sing'

Wie man so im allgemeinen bestimmte TOS-Funktionen "pätscht", wurde in zahlreichen Publikationen bereits demonstriert: man legt den zum TOS-Bereich gehörigen Vektor (z.B. $84 für GEMDOS) auf eine eigene Routine, in der man die auf dem Stack befindliche Funktionsnummer dahingehend untersucht, ob sie mit der zu patchenden übereinstimmt. Ist dem so, ruft man die eigene, ansonsten die Originalroutine auf. Klingt einfach und plausibel? Ist es auch - sofern man Funktionen von GEMDOS, BIOS und XBIOS ändern möchte. Beim Versuch, AES/VDI-Funktionen auf diese Weise zu manipulieren, könnte die Mega-atarianische Selbstmordrate leicht in ungeahnte Höhen steigen: alle Marvins der Galaxis zusammen könnten nicht deprimierter sein als ein programmierender Erdling, der es einfach nicht begreifen kann, warum sein ST Dinge tut, die er eigentlich gar nicht tun dürfte - und so sieht's aus: man biegt den AES/VDI-Trap-Vektor an Adresse $88 auf eine eigene Routine um (wie es oben geschildert wurde)



Außerdem ist es jetzt endlich möglich, zwischen 8x16 und 8x8-Font
mit einer ESC-Sequenz umzuschalten, ohne über das VDI laufen zu müssen!

Der kleine Font eignet sich z.B. hervorragend für Beschriftungen... (Taste)

Bild 2: Umschalten von GEM-Fonts

und ruft probehalber die geänderte Funktion auf. Und siehe da: es funktioniert. Derart begeistert, probiert man's gleich nochmal... Und siehe da: es funktioniert nicht. Aha. Sicherheitshalber Resetknöpfchen drücken und nochmaliges Installieren sind eins; der anschließende Test ergibt das erwartete Ergebnis. Es klappt. Wenn man Glück hat, klappt's sogar noch ein paarmal, um dann aber widersinnigerweise wieder nicht zu klappen, obwohl auch die tausendste Analyse des Listings nicht den Hauch eines Fehlers zutage fördern konnte. Der Ausdruck: "Klappt nicht" ist übrigens dahingehend zu verstehen, daß nicht etwa Bombenteppiche den entgeistert auf den Bildschirm gerichteten Blick weich abzufangen versuchten - nein, das wäre auch zu trivial -, es ist nur einfach so, daß es ganz danach aussieht, als liefe ein total anderes Programm als das eingegebene. Verwirrt? Keine Panik - es gibt eine Lösung. Sie ist eigentlich verblüffend einfach, bloß muß man erst mal draufkommen: durch einen glücklichen Zufall kam heraus, daß der Vektor, der - logisch betrachtet - auf die eigene Routine hätte zeigen müssen, perfiderweis' wieder ins ROM und somit auf die Originalroutinen zeigte!!! Wenn Sie mir jetzt einreden möchten, ich sei von Blind- und/oder Blödheit geschlagen, wenn ich noch nicht mal merken würde, daß nicht mehr meine, sondern die Originalroutinen ausgeführt werden, möchte ich Ihnen zwar nicht grundsätzlich widersprechen, aber zu bedenken geben, daß es sich bei den entsprechenden Routinen um solche handelte, die zum Original kompatibel sind und sich also nicht so besonders gut unterscheiden lassen... Ich hoffe, daß wenigstens den Programmierern von Digital Research der Sinn bekannt ist, warum der GEM-Vektor so ab und an immer mal wieder ins ROM geschubst wird. Daß hierzu auch noch der jedes Programm unglaublich übersichtlich und lesbar machende Line-F-Emulator Verwendung findet, legt die Vermutung nahe, unter den VDI-Programmierern sei ein Sadist gewesen...

Da es außer TRAP #2 keine weitere Schnittstelle zum GEM gibt, mußte der ominöse Pointerverbieger aus GEM selbst kommen; da der Vektor jedoch gar nicht mehr aufs Original zeigte und trotzdem wieder auf die ROM-Adresse umgesprungen wurde, konnte es nur noch eine Interrupt-Routine sein, die da ihr wüstes Unwesen treibt. So ist's denn auch: der erste Slot des Vertical-Blank-Interrupts ist für GEM reserviert, und inmitten dieser IR-Routine residiert der Unhold. Weil ich mich ganz nett über den Typen geärgert hab', verzichtete ich darauf herauszufinden, unter welchen Umständen er in Aktion tritt. Falls es jemand herausgefunden haben sollte (es ist ja durchaus auch eine sehr einfache Erklärung möglich), möge er sich doch bitte melden. Danke.

Die Lösung des Problems war jetzt eigentlich nur noch Formsache und kann in der Cursor-Interrupt-Routine CRS_IRR nachgelesen werden: der Vektor wird auf die eigene Routine gebogen; war er zwischenzeitlich ins ROM gewandert, wird der Cursor ausgeschaltet. Die Praxis hat gezeigt, daß dies so sinnvoll ist; richtig begründen kann ich's aber nicht, weil ich, wie gesagt, keine Lust hatte, in den unergründlichen Tiefen des VDI herumzustöbern (wo Line-F-Emulator und ähnlich gräßliche Dinge zu

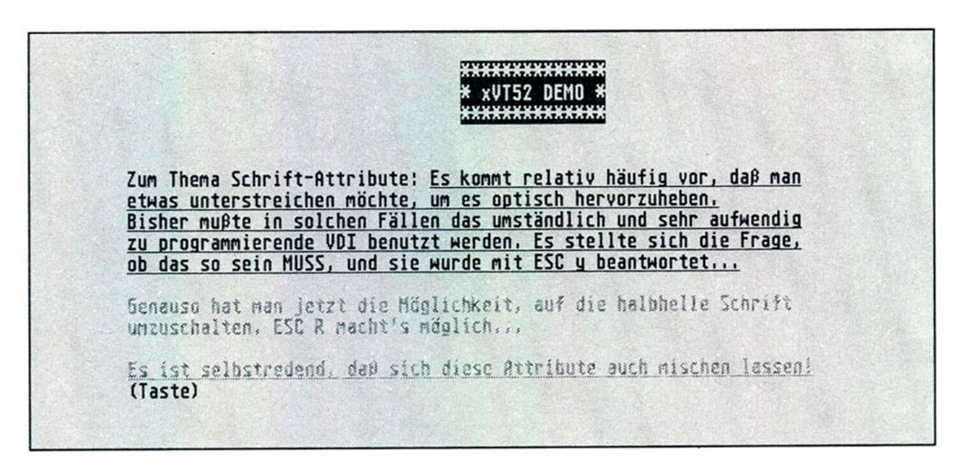


Bild 3: Verschiedene Schriftattribute

Hause sind...). Bemerkenswert ist noch, daß die Vektormanipulation innerhalb der Interrupt-Routine herzerfrischend unanständig ist: wenn GEM auf die hinterlistige Idee kommt, den Vektor ins ROM zu legen, kann es dies nur während des Vertical-Blank-Interrupts in die Tat umsetzen, wobei es die allererste Routine innerhalb der VBL-Schlange benutzt. Die Routine, die den Vektor wieder ins RAM biegt, kommt direkt dahinter zur Ausführung, ohne daß GEM etwas davon wüßte oder es gar verhindern könnte. Ganz schön fies, gell? Und da sich das alles während des VBL-Interruptes abspielt und also zwischenzeitlich kein Programm aktiv sein kann, wird auf diese Tour sichergestellt, daß GEM-Aufrufe im Patch und nicht im ROM landen. Schubidu. Sollten Sie irgendwann einmal in die Verlegenheit kommen, Veränderungen im GEM vorzunehmen (z.B. weil Ihnen die File-Select-Box nicht mehr gefällt oder Sie mehr als vier Fenster haben möchten...), würde ich dringend empfehlen, ebenfalls diese "Technik" zu verwenden; sie hat sich bisher als ebenso problemlos wie zuverlässig erwiesen.

Nachdem der "Brocken" nun aus dem Weg geräumt ist, läßt sich der Rest der Installation wesentlich lockerer nachvollziehen. Als da wäre: Reservieren des benötigten Speicherplatzes, Einfügen der Cursor-Interrupt-Routine in die VBL-Slots sowie Umbiegen der Vektoren für Aufrufe von GEMDOS, AES/VDI, BIOS und XBIOS.

Traphandler

Da - wie bereits erwähnt - alle vier Teile des Betriebssystems über Funktionen zur Textausgabe via VT52 verfügen, mußten dementsprechend auch neue Traphandler geschrieben werden (im Listing tragen sie die ungewöhnlichen Namen GEMDOS, AES_VDI, BIOS und XBIOS). Ihre Lebensaufgabe besteht darin, TOS-Aufrufe dahingehend zu analysieren, ob die angeforderte Funktion zu patchen ist. Wenn ja, werden die Daten- und Adreßregister des Prozessors gerettet, die neugeschriebene Routine angestoßen und die Register wieder restauriert. In allen anderen Fällen wird die Programmkontrolle an den Original-Traphandler weitergegeben, dessen Adresse beim Installieren der eigenen Routinen ja gemerkt wurde. Obwohl durch das Patchen pro Aufruf etwas mehr Code abgearbeitet werden muß, ist keine Verlangsamung des Systems zu befürchten. In der folgenden Übersicht sind alle Funktionen aufgelistet, die von xVT52 abgefangen werden, wobei ein Sternchen darauf hinweist, daß diese Funktionen neu hinzugekommen sind.

GEMDOS:

- 0x2 (CCONOUT): Ausgabe eines Zeichens; Steuerbefehle werden ausgewertet.

- 0x6 (CRAWIO) mit Zusatzparameter <> $FF: wie CCONOUT.

- 0x9 (PRINT LINE): Ausgabe eines mit Null abgeschlossenen Strings, wobei in selbigem enthaltene Steuerzeichen ausgewertet werden.

- 0x40 (WRITE) mit Ausgabekanalnummer 1 (CON:): Ausgabe eines Strings, dessen Länge auf dem Stack übergeben werden muß; Steuerzeichen werden ausgewertet.

VDI:

- 100 (Open Virtual Screen Workstation): Anmelden eines Terminals beim VDI-Server; zusätzlich zur Originalroutine wird der Cursor noch in Home-Position gefahren.

- 101 (Close Virtual Screen Workstation): Abmelden des Bildschirms; zusätzlich zur Originalroutine wird der Cursor ausgeschaltet.

- 3 (Clear Workstation): Bildschirm löschen

- 5 (VDI Escape Sequences), Unterfunktionsnummern:

  - 1 bis 15 (allg. Ausgabe- und Steuerbefehle)

  - 101 (Set Line Offset): Festlegen des Abstandes zwischen log. Bildschirmanfang und 1. Textzeile in Pixel.

  - 102 (Init System Font): Installieren eines Zeichensatzes für normale Textausgaben (ohne Attribute). Diese Funktion beeinflußt NICHT die Textausgabe unter GEM (z.B. VDI 8 und damit letztendlich LINE A-Funktionen).

BIOS:

- 3 (BCONOUT) Zeichenausgabe über die Kanalnummern:

  - 2 (CON:) Ausgabe mit Steuerzeichen

  - 5 (ASC:) Steuerzeichen werden nicht interpretiert, sondern als "normale" Zeichen gedruckt.

* 12 (WRITE ASC) Ausgabe eines mit Null abgeschlossenen Strings, wobei darin enthaltene Steuerzeichen durch entsprechende Symbol ersetzt und nicht interpretiert werden.

XBIOS:

- 21 (CURSCONF) Einstellen der Cursorattribute

PADERCOMP — Walter Ladz

Breslauer Straße 25 · 4790 Paderborn · Tel. (0 52 51) 77 07

Floppystationen

PADERCOMP FL 1
3,5", 1 MB, NEC FD1037 A Laufwerk, anschlußfertig, graues Metallgehäuse, Abm: 153 x 103 x 26 mm **348.00**

PADERCOMP FL 2
Doppellaufwerk übereinander, sonst wie FL 1 . . . . **598.00**

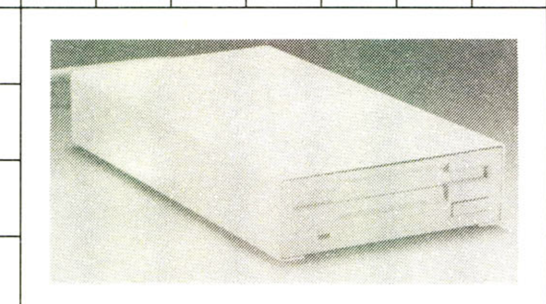
PADERCOMP FL 3
Zweitlaufwerk für Amiga **348.00**

NEC FD 1037 A
3,5", 1 MB, 25,4 mm Bauhöhe, 5V **225.00**

Steckernetzteil
5V/1A, ideal für 2 NEC FD 1037 A **49.00**

Industrie-Floppystecker **7.90**

ST-Kabel an Shugart-Bus 3,5" **29.90**

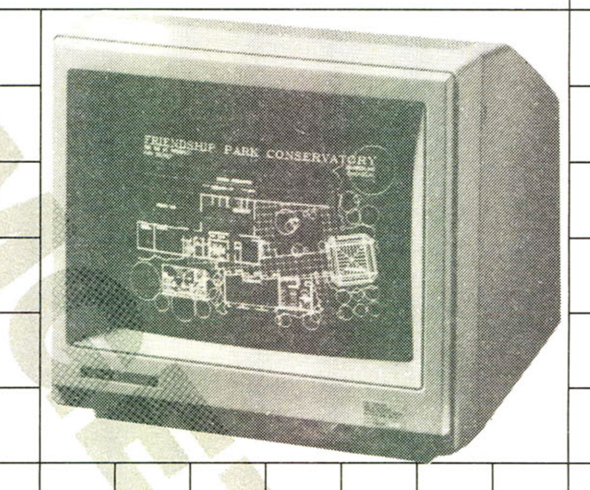


Drucker

NEC P6 24 Nadeln, 216 Z/s, DIN A4 . . . . **1098.–**
NEC P7 24 Nadeln, 216 Z/s, DIN A3 . . . . **1448.–**
NEC P2200 24 Nadeln, P6-kompatibel **948.–**
Star LC-10 9 Nadeln, 144 Z/s, DIN A4 . . . . **598.–**
EPSON LQ-850 24 Nadeln, 264 Z/s, DIN A4 **1398.–**
EPSON LQ-500 24 Nadeln, 180 Z/s, DIN A4 **948.–**

Zubehör

FL-1-Adapter zum Anschluß von 2 FL-1 . . . . **49.00**
Multimatic Cut Sheet Feeder für P6 . . . . **598.00**
Dataphon S 21d-2 300 Baud, FTZ-Nr. **238.00**
Dataphon S 21/23 300 bzw. 1200/75 Baud, BTX . . . **319.00**
Disk Box SS-50 für 50 Stück 3,5"-Disketten **19.90**
Media Box 1 für 150 Stück 3,5"-Disketten **34.90**
Druckerkabel ST **34.90**
Monitor-Ständer dreh-, schwenk- und kippbar **25.00**
Monitor-Stecker für ATARI ST **6.90**
Preisliste . . . . kostenlos



Monitore

EIZO Flexscan 8060 S alle Auflösungen, kein Nachjustieren, 820 x 620 Punkte, 0,28 dot pitch **1498.–**
NEC Multisync alle drei Auflösungen **1398.–**
Mitsubishi Freescan Modell 1471 . . . . **1448.–**
Kabel EIZO/NEC an ATARI ST **75.–**
Monitor-Umschaltbox **47.–**

Die aktuellen Preise erfragen Sie unter der PCB-Mailbox Tel. 0 52 52-34 13 (300 Bd., 8N1)

Bestellungen per Nachnahme ab 30 DM, Auslandslieferungen nur gegen Vorkasse. Eingetragenes Warenzeichen: ATARI ST. Die Preise können günstiger liegen. Rufen Sie an!

Händleranfragen erwünscht!

* 100 (INIT SCREEN) Rücksetzen der Bildschirmausgabe

Wie man sieht, wird die Wichtigkeit der textuellen Ausgabe von TOS mit einem ziemlichen Funktionsaufgebot gebührend gewürdigt und so nimmt es nicht Wunder, daß xVT52 mit seinen über 1800 Zeilen nicht eben zu den Programm-Winzlingen gehört... Die neue Betriebssystemfunktion XBIOS 100 dient dazu, die Ausgabe in einen definierten Zustand zu bringen. Die im Listing mit INIT_CONOUT bezeichnete Routine, die auch mit ESC i aktiviert werden kann, wird während des Installierens aufgerufen und beginnt ihren Dienst mit dem Retten der Zeiger auf die beiden Systemfonts der Kategorie 8x8 und 8x16, wobei zugleich letzterer zum aktuellen Zeichensatz erhoben wird. Der Grund liegt zum einen darin, daß die Fontdaten für die Ausgabe zwingend benötigt werden, andererseits bietet xVT52 auch die Möglichkeit, mit einer einfachen Escape-Sequenz zwischen den beiden Fonts hin- und herzuschalten! Desweiteren werden die Ausgabe-Attribute wie Wrapping, inverse Darstellung, Blinkfrequenz des Cursors etc. auf ihre Defaultwerte gebracht. Vor allem aber wird der "Ausgabevektor" initialisiert. Was es mit diesem auf sich hat, erfahren Sie im folgenden Kapitel.

Vektor-Wirrwarr

Vektoren - neudeutsch Pointer - sind nichts anderes als Speicherbereiche (beim MC68000 sind sie 4 Bytes lang), die eine Adresse enthalten und auf "irgendwas" zeigen. So z.B. auf Variablen, Funktionen oder auch auf weitere Pointer. Die Tatsache, daß man Pointer nicht so einfach verfolgen und manipulieren kann wie normale Variablen, führt gewöhnlich dazu, daß dieses Kapitel beim Erlernen einer Programmiersprache meistens auf dem letzten Platz der Beliebtheitsskala rangiert. Dennoch sind sie absolut unverzichtbar, denn egal, ob man sie in "getarnter" Form (wie in Pascal durch Voransetzen des Schlüsselwortes VAR vor ein Variable oder Prozedur/Funktion innerhalb einer Übergabeliste) oder explizit (wie in C durch den Stern-Operator, z.B. char * string) benutzt: ohne sie geht nichts. Auch nicht bei der Textausgabe. Unter den TOS-Variablen findet man an der Adresse $4A8 einen dieser Zeiger, dessen Zweck mit "Interner Zeiger für Bildschirmausgaberoutinen" angegeben ist. Ich nehme an, daß diese Art kryptischer Erklärungen nicht ganz unschuldig am weitverbreiteten Pointer-Desinteresse ist, weshalb ich ihn etwas genauer beschreiben möchte. Wenn Sie irgendeinen Text auf den Bildschirm ausgeben möchten, rufen Sie hierzu eine TOS-Funktion auf. In dieser wird geprüft, ob der derzeit verwendete Zeichensatz das gewünschte Zeichen enthält. Wenn ja, wird es dargestellt, die Cursorposition erhöht und evtl. der Bildschirminhalt nach oben gescrollt. Damit kann man schon ganz gut leben, bloß muß es auch Möglichkeiten geben, die Ausgabe zu steuern (Bildschirm ganz oder teilweise löschen etc.). Für diese Aufgaben stellt der im TOS vorhandene VT52 Emulator die sog. ESC-Sequenzen zur Verfügung, d.h. durch das ASCII-Zeichen 27 eingeleitete Strings. Für den Emulator bedeutet dies, daß er dieses sowie die nachfolgenden Zeichen nicht einfach ausgeben, sondern interpretieren soll. Dies wird so gelöst, daß die Ausgaberoutine des Betriebssystems nicht direkt angesprungen wird, sondern eben über einen Vektor, der im Normalfall auf diese zeigt. Beim xVT52 Emulator heißt dieser Vektor VEC_BASE und enthält die Adresse der Standard-Ausgaberoutine STD_VEC. Sobald das ESC-Zeichen erkannt wird, wird der Vektor auf eine andere Routine (ESC_SEQ) umgebogen, die dann die gewünschte Funktion in Abhängigkeit des nächsten Zeichens aktiviert. Ohne den "Trick" mit dem Vektor wäre dies -wenn überhaupt- nicht so elegant zu lösen!

In der nächsten Folge werde ich damit beginnen, die neu hinzugekommenen ESC-Sequenzen unter die Lupe zu nehmen und Ihnen erklären, warum die Textausgabe nun viel schneller geworden ist. Die Zwischenzeit können Sie ja mit dem Eintippen des ersten Teils des Listings überbrücken...

MS

```
 1:  ; ************************************
 2:  ; * >EXTENDED VT52-TERMINAL EMULATOR< *
 3:  ; *==================================*
 4:  ; * Entwickelt mit PROFIMAT-Assembler *
 5:  ; *        M.Schumacher, (p) 12/87     *
 6:  ; ************************************
 7:
 8:  LOGBASE  equ $44E     ; Zeiger auf logischen Bildschirm
 9:  VID_R0   equ $FFFF8240   ; Video-Farbregister 0
10:  REGISTER equ d3-d7/a0/a3-a6  ;benötigte Register für
                                   Verschieberoutinen
11:  RESETCO  equ INIT_CONOUT ; Terminal-Initialisierungs-
                               Routine
12:  NVBLS    equ $454 ; Anzahl VBL-Routinen
13:  VBLQUEUE equ $456 ; Zeiger auf Zeiger auf VBL-Routinen
14:
15:   text
16:
17:  INSTALL:
18:   move.l   4(a7),a4  ; ^Basepage
19:   move.l   12(a4),d7 ; Länge TXT
20:   add.l    20(a4),d7 ;       +DTA
21:   add.l    28(a4),d7 ;       +BSS
22:   add.l    #256,d7   ;       +256
23:   move.l   d7,-(a7)  ; = Anzahl zu reservierender Bytes
24:
25:   clr.l    -(a7)       ; SUPER ON
26:   move.w   #$20,-(a7)
27:   trap     #1
28:   move.l   d0,2(a7)    ; alten SSP merken
29:
30:   bsr      INIT_CONOUT  ; Emulator initialisieren
31:   move.l   VBLQUEUE,a0  ; ^VBL-Zeiger
32:   move.w   NVBLS,d0     ; Anzahl der Routinen
33:   subq.w   #1,d0        ; in dbra-Zähler wandeln
34:  \test_slot:
35:   tst.l    (a0)         ; Slot frei?
36:   beq.s    \freeslot    ; ja
37:   addq.l   #4,a0        ; sonst halt
38:   dbra     d0,\test_slot  ; den nächsten Slot testen
39:   bra.s    \noslot      ; kein freier Slot mehr?!
                              (normal unmöglich)
40:  \freeslot:
41:   lea      CRS_IRR,a1   ; Cursor-Interrupt-Routine
```

geht weiter...

```
 42:   move.l    a1,(a0)        ; einbinden
 43:  \noslot:
 44:   lea       TRAPS,a4       ; ^Speicher für Original-Trap-
                                  Handler
 45:   move.l    $84,(a4)+      ; alten GEMDOS-Vektor merken
 46:   lea       GEMDOS,a2      ; und neuen
 47:   move.l    a2,$84         ; installieren
 48:   move.l    $88,(a4)+      ; dasselbe mit GEM-Vektor
 49:   lea       AES_VDI,a2
 50:   move.l    a2,$88
 51:   move.l    $B4,(a4)+      ; BIOS-Vektor
 52:   lea       BIOS,a2
 53:   move.l    a2,$B4
 54:   move.l    $B8,(a4)       ; XBIOS-Vektor
 55:   lea       XBIOS,a2
 56:   move.l    a2,$B8
 57:
 58:   trap      #1             ; SUPER OFF
 59:   addq.l    #6,a7          ; Stack aufräumen
 60:
 61:   move.w    #$31,-(a7)     ; KEEP PROCESS (hi zombie!)
 62:   trap      #1             ; auf Wiedersehen im GEMDOS
 63:  ;****************************************
 64:  ;* G E M D O S / A E S+V D I / B I O S / *
 65:  ;* X B I O S - TRAPHANDLER               *
 66:  ;****************************************
 67:
 68:  SAVE_REGS:                   ; Register retten
 69:   move.l    (a7)+,d0         ; Rücksprungadresse retten
 70:   lea       SAVE_TOP,a1      ; A1=Registerspeicher,
                                    Obergrenze
 71:   move.w    (a7)+,-(a1)      ; save(SR)
 72:   move.l    (a7)+,-(a1)      ; save(PC)
 73:   movem.l   d3-d7/a3-a7,-(a1) ;save(Register)
 74:   lea       2(a0),a7         ; SSP auf 1. Parameter setzen
 75:   move.l    d0,-(a7)         ; Rücksprungadresse zurück
 76:   rts
 77:
 78:  REST_REGS:                   ; Register zurückholen
 79:   lea       SAVE_BOT,a1      ; A1=Registerspeicher,
                                    Untergrenze
 80:   movem.l   (a1)+,d3-d7/a3-a7 ;Register zurück
 81:   move.l    (a1)+,-(a7) ; PC zurück
 82:   move.w    (a1)+,-(a7) ; SR zurück
 83:   rte                     ; zurück ins aufrufende Programm
 84:
 85:  SAVE_BOT:                 ; Untergrenze Zwischenspeicher
 86:   ds.w      23,0           ; Platz für Register, PC und SR
 87:  SAVE_TOP:                 ; Obergrenze Zwischenspeicher
 88:
 89:
 90:  GEMDOS:
 91:   movea.l   a7,a0        ; A0=SSP;
 92:   btst      #5,(a0)      ; if (Aufruf aus S-Mode)
 93:   beq.s     \from_user
 94:   addq.l    #6,a0        ; then Offset addieren
 95:   bra.s     \test_pline ; (PC.L+SR.W)
 96:  \from_user:
 97:   move.l    USP,a0       ; else User Stack benutzen;
 98:  \test_pline:
 99:   cmpi.w    #9,(a0)      ; if (Funktion == PRINT LINE)
100:   beq.s     \pline       ; then eigene Funktion benutzen
101:   cmpi.w    #64,(a0)     ; else if (Funktion == WRITE)
102:   beq.s     \write       ; then auf Ausgabe-Kanal prüfen
103:   cmpi.w    #2,(a0)      ; else if (Funktion == CCONOUT)
104:   beq.s     \cconout     ; then eigene Funktion benutzen
105:   cmpi.w    #6,(a0)      ; else if (Funktion != CRAWIO)
106:   bne.s     \orig        ; then Originalroutine rufen
107:   cmpi.b    #-1,3(a0)    ; else if (Zeichen == 255)
108:   beq.s     \orig        ; then Original aufrufen
109:  \cconout:
110:   bsr       SAVE_REGS    ; Register retten
111:   bsr       CON_OUT      ; Zeichen ausgeben
112:   bra       REST_REGS    ; Register zurück, fertig
113:  \pline:
114:   bsr       SAVE_REGS'   ; Register retten
115:   bsr       WRITE        ; String ausgeben
116:   bra       REST_REGS    ; Register zurück, fertig
117:  \orig:
118:   move.l    TRAPS,a0     ; else Originalroutine
119:   jmp       (a0)         ; benutzen
120:  \write:
121:   cmpi.w    #1,2(a0)     ; Ausgabe auf CON:?
122:   bne.s     \orig        ; nein, ham wa nix mit zu tun
123:   addq.w    #2,a0        ; sonst Anzahl auszugebender
124:   move.l    2(a0),d1     ; Zeichen holen
125:   beq.s     \retour      ; kein Zeichen ist definitiv zu
                                wenig!
126:   bsr       SAVE_REGS  ; ansonsten Register retten
127:   addq.w    #4,a7        ; (a7) = String-Pointer
128:   move.l    d1,-(a7)     ; Anzahl auf Stack retten
129:   move.l    4(a7),-(a7) ; String-Pointer auf Stack legen
130:  \lp:
131:   move.l    (a7),a0 ; String-Pointer holen
132:   addq.l    #1,(a7) ; auf nächstes Zeichen zeigen lassen
133:   move.b    (a0),d1 ; Zeichen holen
134:   and.w     #$FF,d1 ; nur LSB beachten
135:   move.w    d1,-(a7) ; Zeichen auf Stack legen
136:   bsr       CON_OUT  ; und ausgeben
137:   addq.w    #2,a7     ; Stack korrigieren
138:   subq.l    #1,4(a7) ; Anzahl Zeichen dekrementieren
139:   bne.s     \lp       ; und nächstes Zeichen ausgeben
140:   addq.l    #8,a7     ; Zähler und Pointer löschen
141:   bra       REST_REGS   ; Register restaurieren und
                              zurückspringen
142:  \retour:
143:   rte
144:
145:
146:  AES_VDI:
147:   cmpi.w    #$73,d0     ; VDI-Aufruf?
148:   bne.s     \orig       ; nein, AES
149:   move.l    a0,-(a7)    ; a0 retten
150:   move.l    d1,a0       ; ^ParameterBlock
151:   move.l    (a0),a0     ; ^ContrlBlock
152:   cmpi.w    #5,(a0)     ; VDI-ESC-Sequenz? (CONTRL[0])
153:   bne.s     \op_close ; nein
154:   lea       10(a0),a0 ; ^Unterfunktionsnummer (CONTRL[5]
155:   cmpi.w    #16,(a0)  ; <16?
156:   bmi.s     \ok         ; ja, eigene Routine ausführen
157:   cmpi.w    #101,(a0) ; VDI ESC 101?
158:   beq.s     \ok         ; ja
159:   cmpi.w    #102,(a0) ; Font installieren?
160:   bne.s     \fail   ; nein, dann Originalroutine benutzen
161:  \ok:              ; *** eigene Routinen ausführen ***
162:   lea       VDI_TOP,a0     ; ^Zwischenspeicher
163:   move.l    (a7)+,-(a0)    ; alten Inhalt von a0 retten
164:   move.w    (a7)+,-(a0)    ; SR
165:   move.l    (a7)+,-(a0)          ; PC
166:   movem.l   d0-d7/a1-a7,-(a0)   ;alle Register retten
167:   bsr       VDI_ENTRY            ; Routine ausführen
168:   lea       VDI_BOT,a0           ; ^Zwischenspeicher
169:   movem.l   (a0)+,d0-d7/a1-a7   ;Register wieder zurück
170:   move.l    (a0)+,-(a7)          ; PC,
171:   move.w    (a0)+,-(a7)          ; SR und
172:   move.l    (a0)+,a0             ; a0 restaurieren,
173:   rte                            ; Exception beenden
174:  \fail:
175:   move.l    (a7)+,a0             ; a0 restaurieren
176:  \orig:
177:   move.l    TRAPS+4,-(a7)        ; Originaladresse AES/VDI
178:   rts                            ; benutzen
179:  \op_close:
180:   cmpi.w    #100,(a0)      ; OpenVirtualScreenWorkstation?
181:   bne.s     \close         ; nein
182:   lea       TCB,a0         ; ^TerminalControlBlock
183:   clr.l     (a0)           ; Zeile und Spalte auf 0
184:   bra.s     \fail          ; zusätzlich Originalroutine
                                  ausführen
185:  \close:
186:   cmpi.w    #101,(a0)   ; CloseVirtualScreenWorkstation?
187:   bne.s     \clear      ; nein
188:   lea       CCB,a0      ; ^CursorControlBlock
189:   bclr      #3,(a0)     ; Cursor abschalten
190:   bra.s     \fail       ; und Originalroutine ausführen
191:  \clear:
192:   cmpi.w    #3,(a0)     ; ClearWorkstation?
193:   bne.s     \fail       ; nein
194:   movem.l   d0-d7/a1-a5,-(a7) ; Register retten
195:   lea       TCB,a0            ; ^TerminalControlBlock
196:   bsr       CLS               ; Bildschirm löschen
197:   movem.l   (a7)+,d0-d7/a1-a5 ; Register zurück
198:   bra.s     \fail             ; zusätzlich Original-
199:                                 ; routine ausführen
200:  VDI_BOT:
201:   ds.w      35,0              ; Platz für d0-a7, SR & PC
202:  VDI_TOP:
203:
204:
205:  BIOS:
206:   movea.l   a7,a0             ; A0=SSP;
207:   btst      #5,(a0)           ; if (Aufruf aus S-Mode)
208:   beq.s     \from_user
209:   addq.l    #6,a0             ; then Offset addieren
```

geht weiter...

```
210:   bra.s     \test_wr_asc     ; (PC.L+SR.W)
211:  \from_user:
212:   move.l    USP,a0           ; else User Stack benutzen;
213:  \test_wr_asc:
214:   cmpi.w    #12,(a0)         ; if (Funktion != WRITE_ASC)
215:   bne.s     \test_bconout    ; then auf BCONOUT testen
216:   bsr       SAVE_REGS        ; else { Register retten;
217:   bsr       WRITE_ASC        ; String ausgeben;
218:   bra       REST_REGS        ; Register zurück; beenden }
219:  \test_bconout:
220:   cmpi.w    #3,(a0)          ; if (Funktion == BCONOUT)
221:   beq.s     \test_dev        ; then auf Ausgabegerät
                                    testen
222:  \orig_bios:
223:   move.l    TRAPS+8,-(a7)    ; else Originalroutine benutzen
224:   rts
225:  \test_dev:
226:   addq.w    #2,a0            ; Zeiger auf Gerätenummer
                                    setzen
227:   cmpi.w    #2,(a0)          ; if (Ausgabegerät != Console)
228:   bne.s     \test_dev_vid    ; then auf ASCII-Ausgabe
                                    testen
229:   bsr       SAVE_REGS        ; else { Register retten;
230:   bsr       CON_OUT          ; Zeichen ausgeben;
231:   bra       REST_REGS        ; Register zurück; beenden }
232:  \test_dev_vid:
233:   cmpi.w    #5,(a0)          ; if (!ASCII_Ausgabe)
234:   bne.s     \orig_bios       ; then Originalroutine benutzen
235:   bsr       SAVE_REGS        ; else { Register retten;
236:   bsr       ASC_OUT          ; Zeichen ausgeben;
237:   bra       REST_REGS        ; Register zurück; beenden }
238:
239:
240:  XBIOS:
241:   movea.l   a7,a0            ; A0=SSP;
242:   btst      #5,(a0)          ; if (Aufruf aus S-Mode)
243:   beq.s     \from_user
244:   addq.l    #6,a0            ; then Offset addieren (PC.L+SR.W)
245:   bra.s     \test_cursconf
246:  \from_user:
247:   move.l    USP,a0           ; else User Stack benutzen;
248:  \test_cursconf:
249:   cmpi.w    #21,(a0)         ; if (Funktion == CURSCONF)
250:   beq.s     \cursconf        ; then eigene Funktion benutzen
251:   cmpi.w    #100,(a0)        ; else if (Funktion
                                    == INIT_SCREEN)
252:   beq.s     \init            ; then Bildschirmausgabe
                                    initialisieren
253:   move.l    TRAPS+12,-(a7)   ; else Originalroutine
                                    benutzen
254:   rts
255:  \cursconf:
256:   bsr       SAVE_REGS        ; Register retten
257:   bsr       CURSCONF         ; eigene Routine ansto_en
258:   bra       REST_REGS        ; Register zurück, beenden
259:  \init:
260:   bsr       SAVE_REGS        ; Register retten
261:   bsr.s     INIT_CONOUT      ; Ausgabe initialisieren
262:   bra       REST_REGS        ; Register zurück, beenden
263:
264:
265:  ;**********************************************
266:  ;* NEUE  ROUTINEN  FÜR  VT52  UND  VDI-ESCAPES *
267:  ;**********************************************
268:
269:  INIT_CONOUT:              ; ESC 'i', Terminal initialisieren
270:   dc.w      $A000          ; Line A-Pointer holen
271:   lea       TCB,a2         ; TerminalControlBlock
                                  initialisieren
272:   move.l    4(a1),a4         ; ^8*8-Font
273:   move.l    $4C(a4),28(a2)   ; ^Fontdaten speichern
274:   move.l    8(a1),a4         ; ^8*16-Font
275:   move.l    $4C(a4),a4       ; ^Fontdaten
276:   move.l    a4,24(a2)        ; speichern
277:   move.l    a4,32(a2)        ; und 8x16 zum aktuellen Font
                                    machen
278:   move.l    #$4f0018,4(a2)   ; max.Spalte=79, max.Zeile=24
279:   move.b    #2,8(a2)         ; Wrapping ein-, inverse
                                    Darstellung ausschalten
280:   clr.w     10(a2)           ; keine Cursor-Position
                                    gespeichert (ESC 'j')
281:   move.l    #$100500,36(a2)  ; Zeichenhöhe 16 Pixels,
                                    16*80 Bytes/Textzeile
282:
283:   lea       LINE_A,a2        ; Line-A Adresse
284:   move.l    a0,(a2)          ; merken
285:
286:   lea       VEC_BASE,a2      ; Ausgabevektor initialisieren
287:   lea       STD_VEC,a3       ; ^Standardausgabe
288:   move.l    a3,(a2)          ; als Default übernehmen
289:
290:   lea       CCB,a4           ; CursorControlBlock
                                    initialisieren
291:   move.w    #$200,(a4)       ; Cursor ausschalten,
                                    Blinkmodus wählen
292:   move.l    #$140014,2(a4)   ; Blinkrate und -zähler auf
                                    20 stellen
293:   move.l    LOGBASE,6(a4)    ; abs. Cursorposition auf
                                    Bildschirm-Anfang
294:
295:   lea       BEL_ADR,a0       ; ^Speicher für Adresse
                                    Gong-Routine
296:   move.l    #$FC201C,d0      ; TOS-Adresse (Version vom
                                    06.02.1986)
297:   cmpi.w    #$1986,$FC001A   ; "altes" TOS?
298:   beq.s     \old_tos         ; ja
299:   move.l    #$FC2270,d0      ; sonst Blitter-TOS-Adresse
                                    nehmen
300:  \old_tos:
301:   move.l    d0,(a0)          ; Vektor installieren
302:
303:   lea       TCB,a0           ; ^TerminalControlBlock
304:   bsr       CLS              ; Bildschirm löschen+Home
305:   bsr       DEF_TABS         ; Tabulatoren auf default
setzen
306:   bra       UPDATE_END       ; Cursor freigeben und zurück
307:
308:
309:  CRS_IRR:                     ; Cursor-Interruptroutine
310:   move.l    $88,d1           ; derzeit aktive AES/VDI-Adresse
311:   lea       AES_VDI,a0       ; eigenen Trap-Handler neu
                                    installieren
312:   move.l    a0,$88           ; AES/VDI überlisten (grins)
313:   cmp.l     a0,d1            ; hat sich Adresse geändert?
314:   bne       UPDATE_CRS       ; dann Cursor abschalten
315:   lea       CCB,a4           ; ^CursorControlBlock
316:   btst      #3,(a4)          ; Cursor eingeschaltet?
317:   beq.s     \irr_end         ; nein, fertig
318:   btst      #0,(a4)          ; wird gerade Text ausgegeben?
319:   beq.s     \irr_end         ; ja, dann nicht stören
320:   btst      #1,(a4)          ; Blinken eingeschaltet?
321:   bne.s     \blink           ; ja
322:   btst      #2,(a4)          ; Cursorposition schon
                                    invertiert?
323:   bne.s     \irr_end         ; ja, nicht mehr invertieren
324:  \inv:
325:   bsr       CUR_INV          ; Cursorposition invertieren
326:  \irr_end:
327:   rts                        ; zurück zum IR-Slot-Handler
328:  \blink:
329:   subq.w    #1,4(a4)         ; Blinkzähler -
330:   bne.s     \irr_end         ; noch nicht 0, fertig
331:   move.w    2(a4),4(a4)      ; sonst Zähler neu laden
332:   bra.s     \inv             ; invertieren und zurück
333:
334:
335:  CURSCONF:                    ; XBIOS(21)
336:   bsr       UPDATE_CRS       ; Cursor abschalten
337:   move.w    4(a7),d1         ; Funktionsnummer holen
338:   bne.s     \f1              ; nicht 0
339:   bclr      #3,(a4)          ; Cursor ausschalten
340:   bra.s     \zurück          ; fertig
341:  \f1:
342:   cmpi.w    #1,d1            ; Fkt.-Nr. 1?
343:   bne.s     \f2              ; nein
344:   bset      #3,(a4)          ; Cursor einschalten
345:   bra.s     \zurück
346:  \f2:
347:   cmpi.w    #2,d1            ; Fkt.-Nr. 2?
348:   bne.s     \f3              ; nein
349:   bset      #1,(a4)          ; Cursor in Blinkmodus versetzen
350:   bra.s     \zurück
351:  \f3:
352:   cmpi.w    #3,d1            ; Fkt.-Nr. 3?
353:   bne.s     \f4              ; nein
354:   bclr      #1,(a4)          ; Blinkmodus ausschalten
355:   bra.s     \zurück
356:  \f4:
357:   cmpi.w    #4,d1            ; Fkt.-Nr. 4?
358:   bne.s     \f5              ; nein
359:   move.w    6(a7),2(a4)      ; Blinkrate setzen
360:   move.w    6(a7),4(a4)      ; Zähler initialisieren
361:   bra.s     \zurück
362:  \f5:
```

geht weiter...

```
363:   cmpi.w   #5,d1       ; Fkt.-Nr. 5?
364:   bne.s    \zurück     ; nein, ignorieren
365:   clr.l    d0          ; d0.L wegen Wortoperation löschen
366:   move.w   2(a4),d0    ; Blinkrate holen
367: \zurück:
368:   move.l   d0,-(a7)     ; Rückgaberegister retten
369:   bsr      UPDATE_END   ; Cursor wieder freigeben
370:   move.l   (a7)+,d0     ; Register zurück
371:   rts
372:
373:
374: WRITE_ASC:                  ; String ausgeben (ohne
                                  Steuerzeichen)
375:   move.l   4(a7),a6      ; ^Ausgabestring
376: \lp:
377:   move.b   (a6)+,d0      ; auszugebendes Byte holen
378:   beq.s    \ende         ; falls 0: fertig
379:   bsr.s    ASC_ENTRY     ; ausgeben
380:   bra.s    \lp           ; bis String-Ende
381: \ende:
382:   rts
383:
384: ASC_OUT:                ; DIREKTE ZEICHENAUSGABE(0x00-0xFF)
385:   move.w   4(a7),d0 ; Zeichen holen
386: ASC_ENTRY:
387:   andi.w   #$FF,d0  ; nur Byte beachten
388:   bra      PUT      ; und ausgeben
389:
390:
391: WRITE:                  ; String ausgeben
                              (mit Steuerzeichen)
392:   move.l   4(a7),a6 ; ^Ausgabestring
393: \lp:
394:   move.b   (a6)+,d1 ; auszugebendes Byte holen
395:   bne.s    \aus     ; ausgeben, falls<>0
396:   lea      TCB+9,a0 ; ^Grafik-Flag
397:   tst.b    (a0)     ; Grafik eingeschaltet?
398:   beq.s    \ende    ; nein, fertig
399: \aus:
400:   move.b   d1,d0    ; Zeichen ausgeben
401:   bsr.s    CON_ENTRY
402:   bra.s    \lp      ; bis String-Ende
403: \ende:
404:   rts
405:
406: CON_OUT:                     ; AUSGABE MIT STEUERZEICHEN
407:   move.w   4(a7),d0        ; Zeichen holen
408: CON_ENTRY:
409:   andi.w   #$FF,d0         ; LSB isolieren
410:   lea      VEC_BASE,a1     ; ^Vektor
411:   move.l   (a1),a0         ; Vektor holen
412:   jmp      (a0)            ; und anspringen
413:
414:
415: STD_VEC:
416:   cmpi.w   #" ",d0     ; Zeichen < Blank?
417:   bpl      PUT         ; nein, ausgeben
418:   cmpi.w   #$1B,d0     ; ESC?
419:   bne.s    \control    ; nein
420:   lea      ESC_SEQ,a0  ; ^ESC-Handler
421:   move.l   a0,(a1)     ; als Vektor für nächstes
                             Zeichen merken
422: \return:
423:   rts                  ; fertig
424: \control:
425:   subq.w   #7,d0          ; <7?
426:   bmi.s    \return        ; wenn ja, dann nicht beachten
427:   cmpi.w   #7,d0          ; >13?
428:   bpl.s    \return        ; ja: Ausgabe unterdrücken
429:   lea      BEL,a1         ; ^dingelingeling
430:   add.w    d0,d0          ; Zeichenoffset in Word-Pointer
                                umwandeln
431:   move.w   CTRL(PC,d0.w),a2  ; entsprechenden Vektor
432:   add.l    a1,a2           ; +Offset 1. Routine
433:   lea      TCB,a0          ; ^TerminalControlBlock
434:   bsr.s    UPDATE_CRS      ; Cursor abschalten
435:   jsr      (a2)            ; ausführen
436:   bra.s    UPDATE_END      ; Cursor wieder einschalten
437:
438:
439: CTRL:     dc.w  0          ; Glocke
440:           dc.w  BS-BEL     ; Backspace
441:           dc.w  TAB-BEL    ; Tabulator
442:           dc.w  LF-BEL     ; Zeilenvorschub
443:           dc.w  LF-BEL     ; Vertikaltabulator (wie
                                 Zeilenvorschub)
444:           dc.w  LF-BEL     ; Formularvorschub  (wie
                                 Zeilenvorschub)
445:           dc.w  CR-BEL     ; Wagenrücklauf
446:
447:
448: UPDATE_CRS:              ; Cursor für Textausgabe
                               ausschalten
449:   lea      CCB,a4      ; ^CursorControlBlock
450:   bclr     #0,(a4)     ; disable Cursor
451:   btst     #2,(a4)     ; Cursor sichtbar?
452:   beq.s    \return     ; nein
453:   bra      CUR_INV     ; sonst Cursorposition invertieren
454: \return:
455:   rts
456:
457: UPDATE_END:                  ; Cursor wieder freigeben
458:   lea      TCB,a0          ; ^TerminalControlBlock
459:   move.l   LOGBASE,a1      ; ^Video-RAM
460:   move.w   38(a0),d0       ; Bytes/Textzeile
461:   mulu     2(a0),d0        ; * akt. Zeile
462:   adda.l   d0,a1           ; + Bildschirm-Anfang
463:   adda.w   (a0),a1         ; + akt. Spalte
464:   move.l   a1,6(a4)        ; = abs. Cursorposition
465:   bset     #0,(a4)         ; enable Cursor
466:   rts
467:
468:
469: ESC_SEQ:                     ; Esc-Sequenz verarbeiten;
                                   Einsprung mit:
470:                              ; d0.w = Steuerzeichen;
                                   a1 = ^VEC_BASE
471:   lea      VEC_BASE,a1 ; Ausgabe-Vektor
472:   lea      STD_VEC,a0  ; wieder auf die normale
                             Ausgabe biegen
473:   move.l   a0,(a1)         ; umschalten
474:   subi.w   #"A",d0         ; Offset abziehen
475:   bmi.s    \esc_fail       ; zu klein, ungültig
476:   cmpi.w   #"Z"-"A",d0     ; <="Z"?
477:   ble.s    \esc_big        ; ja, entsprechende Routine
                                 ausführen
478:   bra.s    \esc_sml        ; sonst auf Kleinbuchstaben
                                 testen
479: \esc_fail:
480:   rts
481: \esc_big:
482:   bsr.s    UPDATE_CRS      ; Cursor ausschalten
483:   lea      CRS_UP,a1       ; Adresse der 1. Routine
484:   add.w    d0,d0           ; Zeichen als Wort-Offset
485:   lea      E_BIG,a2        ; Adress-Tabelle
486:   move.w   0(a2,d0.w),a2   ; Adress-Offset holen
487:   adda.l   a1,a2           ; + 1. Routine
488:   lea      TCB,a0          ; ^TerminalControlBlock
489:   jsr      (a2)            ; Routine ausführen
490:   bsr.s    UPDATE_END      ; Cursor wieder ein
491:   rts                      ; fertig
492: \esc_sml:
493:   subi.w   #"a"-"A",d0     ; Offset für Kleinbuchstaben
                                 abziehen
494:   bmi.s    \esc_fail       ; <"a", ignorieren
495:   cmpi.w   #"z"-"a",d0     ; >"z"?
496:   bgt.s    \esc_fail       ; ja, ignorieren
497:   bsr      UPDATE_CRS      ; Cursor ausschalten
498:   lea      SET_COLOR,a1    ; Adresse der 1. Routine
499:   add.w    d0,d0           ; Zeichen als Wort-Offset
500:   lea      E_SML,a2        ; ^Adress-Tabelle
501:   move.w   0(a2,d0.w),a2   ; Adress-Offset holen
502:   adda.l   a1,a2           ; + 1. Routine
503:   lea      TCB,a0          ; ^TerminalControlBlock
504:   jsr      (a2)            ; Routine ausführen
505:   bsr      UPDATE_END      ; Cursor einschalten
506:   rts                      ; fertig
```

ENDE

1ST-ADDRESS

die schnellste Dateiverwaltung für den ATARI-ST
zur professionellen Kunden-, Mitglieder-, Artikel-, Dokumenten-Verwaltung

- läuft als Accessory im Hintergrund; volle GEM-Einbindung
- dank Assembler-Programmierung und Steckmodul-Technik extrem schnell:
  — Scrollen: 52 Zeilen pro Sekunde
  — Suchen: aus 6000 Datensätzen in 1 Sek.
  — Sortieren: 6000 Datensätze in 7,5 Sek.
- freie Gestaltung der Datenfelder sowie der Bildschirmmasken, Etiketten und Listenausdrucke
- kann bis zu 6 Dateien gleichzeitig verwalten, und zwar je nach Hauptspeicher- und Laufwerks-Größe 7000 bis 40000 Datensätze
- Suchfunktion mit vielfältigen Selektionskriterien; Sortierfunktion auf alle Felder anwendbar
- Import-/Export-Funktion mit Zugriff auf alle Laufwerke und Schnittstellen
- automatisches Anwählen von Telefon/Telex/Telefax über Modem möglich
- in Verbindung mit Textprogrammen (1ST-WORD/1ST-MAIL, WordPerfect u. a.) automatische Serienbrieferstellung; Programmierschnittstelle
- für alle ATARI ST

DM 148,- inkl. MwSt.

Bei Vorkasse (Scheck) keine Versandkosten;
Demo-Diskette mit Handbuch DM 10,-
Komplettes Programm für 14 Tage zum Testen DM 20,-

VICTOR KG SOFTWARE-ENTWICKLUNG
Postfach 1510 - D-2058 Lauenburg - Tel. (0 41 53) 5 23 23





Hardware-Software Systemlösungen
Harmel-Scollar & Schmithals GbR
Darmstädter Str. 20 · 5000 Köln 1 · ☎ 0221/316207

Ihr Spezialist für ATARI Computer
Planung ... Anpassungen ... Schulung ... Service

Versandbedingungen: Preise incl. Versandkosten (Scheck), bei Nachnahme zuzügl. DM 5.–, alle Preise für Atari St-Version.

Der Partner für Handwerk- und Kleinbetriebe. Preiswerte und maßgeschneiderte EDV-Systeme durch unsere unabhängige Beratung.

Bavaria Soft Handwerkspaket 2298,–
BS-Handel 898,–
Prodata fibuMAN e.........................389,–
Cash Soft T.I.M. (Fibu) 259,–
Cash Flow (Kassenbuch) 259,–
Depot (Faktura) 479,–

Megamax C/ Megamax Modula 2 .... je 369,–
GFA Basic / Compiler .... je 88,–
GFA Draft Plus.......................................298,–
PCB Platinen Layout.............................. 189,–
Stad (Zeichenpr. mit 3 D Teil)...........................155,–
Star Writer St 189,–
Signum II 369,–
Tempus 2.0..95,–
1st Word Plus/Adimens ST/Talk......... je 189,–
Harddisk Utility (Harddisk Backup) 49,–
Vortex HDPLUS 20 / 30 .... 1168,–/1348,–
und ... und ... und ...
Preisliste anfordern





In der diesmaligen Folge unserer Programmierpraxis geht es wieder voll durch alle Sprachen durch. Alle ist natürlich etwas übertrieben, aber für die Freunde von C, Assembler und BASIC ist diesmal etwas dabei.

Mit Bug-Alert lassen sich Fehler in GEM-Programmen entfernen, an die man sich sonst nur mühsam über printf- oder scanf-Aufrufe herantasten konnte. Man kann sich nun mittels einer Alert-Box durch das fehlerhafte Programm tracen.
"Hergehört", heißt es, wenn die Diskette, die soeben ins Laufwerk geschoben wurde, nicht schreibgeschützt ist. Mit diesem kleinen Programm im Auto-Ordner wird es wirklich schwer, aus Versehen irgendeine Datei zu löschen.
Fractals dürften den meisten Leuten ein Begriff sein. Wie man sie programmiert und dann die hübschen Apfelmännchen, Mandelbrot- und Julia-Mengen erhält, wird hier beschrieben.

I N H A L T



BUG ALERT

ein Mini-Tool zum Entwanzen von C-Programmen unter GEM

Die Sprache C läßt dem Programmierer bei der Arbeit viel Freiheit - leider auch die Freiheit, Fehler zu machen. Ein großer Anteil der Programmierarbeit besteht so in der Fehlersuche. Arbeitet man mit einem Compiler, der über die normale Syntax-Prüfung hinaus wenig Unterstützung bei der Fehlersuche bietet, ist man darauf angewiesen, sich mit Hilfe von 'printf'- und 'scanf'- Aufrufen, mühselig an den Fehler heranzutasten.

Sehr problematisch wird diese Methode aber, wenn man unter GEM programmiert. Die eingestreuten printf-Befehle bringen den 'Schreibtisch' heillos durcheinander, und man tappt dann mit der Maus auf einem weißen Monitor herum, um die Programmknöpfe zu finden!

Nachdem mir diese Methode mehrfach die ganze Menüleiste zerstört hatte, überlegte ich mir, daß es wohl sinnvoller sei, den Problemen eines GEM-Programms mit den Mitteln zu begegnen, die GEM zur Verfügung stellt, nämlich den Dialogboxen. Diese haben ja den Sinn, dem Programmbenutzer Informationen über den augenblicklichen Zustand des Programms zu übermitteln (Helptexte, Fehlermeldungen).

Anfangs verstreute ich die Alertboxen in der Nähe des vermuteten Fehlers direkt im Programm und entfernte sie nach dem Auffinden des Fehlers wieder. Als ich aber dann in einem besonders vertrackten Modul die Alertboxen mehrfach eingesetzt und wieder abgeräumt hatte, kam mir die Idee, sie doch zunächst einmal im Programm zu lassen und über eine globale Boolsche Variable 'tr' einen Trace - Modus zu simulieren.

Im kritischen Programmbereich haben die Funktionen dann folgendes Aussehen:

```
x_funktion()
{
if(tr) gprintf("x_funktion"); ...
...
...
}
```

oder:

```
x_funktion(arg1,arg2)
int   arg1;
float arg2;
{
if(tr) gprintf("x_funktion|arg1 =
%darg2 = %2f", arg1, arg2)
...
...
...
}
```

Die Funktion meldet sich dann bei eingeschaltetem Trace-Modus mit:

Alertbox 1

```
x_funktion
arg1 = 1
arg2 = 2
[ABBRUCH] [TROFF] [OK]
```

Bei diesem Vorgehen ist nach dem ersten Durchlauf klar, in welcher Funktion der Wurm steckt. Eine genaue Überprüfung dieser Funktion nach folgender Checkliste führt oft schon an dieser Stelle zum Finden des Fehlers:

1. Werden die GEM-Funktionen mit der richtigen Zahl von Parametern aufgerufen ??
2. Stimmen die Typen der übergebenen Parameter mit den von der Funktion erwarteten überein ??
3. Semikolon hinter 'for', 'while' oder 'if' - Klammer ??
4. Zuweisungsoperator '=' wo eigentlich '==' stehen sollte ?
5. Hat sich ein Zeichen (z.B. ';') ganz weit an den rechten Rand des Listings verirrt, so daß es beim normalen Durchscrollen des Quelltextes nicht zu erkennen ist??

Falls die Überprüfung dieser fünf Punkte keinen Erfolg bringt, sollte man sich als nächstes die folgenden Fragen vorlegen:

6. Sind alle Pointer richtig initialisiert ?
7. Sind alle Datenstrukturen richtig initialisiert ?

Zur Beantwortung dieser Fragen bieten sich nun die Funktionen an, um die es in diesem Artikel geht: 'gprintf' und 'dump':

Mit gprintf kann man sich den jeweils aktuellen Wert der Variablen in jedem Format ausdrucken lassen, das vom normalen 'printf'-Befehl erzeugt werden kann - dump erzeugt einen Hex-Dump von 4 mal 8 Bytes ab der angegebenen Speicheradresse. Den Aufruf dieser Funktionen macht man nun besser nicht von 'tr' abhängig, denn da man zu diesem Zeitpunkt

die Lage des Fehlers kennt, wäre es sinnlose Zeitvergeudung, sich im Trace-Modus bis zur fraglichen Funktion durchzuklicken!

Angenommen 'gprintf' oder 'dump' zeigen für einen Pointer den Wert Null an. Nun ist es natürlich sinnvoll, das Programm an dieser Stelle zu unterbrechen - denn ein Nullpointer in Aktion macht den Griff zum Resetknopf unausweichlich. - Und wenn auch die benutzte Ramdisk resetfest ist, der Zeitverlust gegenüber einem geordneten Programmabbruch ist doch erheblich: Abfragedialog, ob Ramdisk so bleiben soll. Wiederaufbau des Desktops, Anzeige der Directories, Neustarten der Shell.. also es ist schon sinnvoll zu programmieren:

```
x_funktion()
{
int var;
...
...
if(!gprintf("var = %d",var))
    return;
...
}
```

Falls ein Fehler behoben ist, kann man sich dann durch Anklicken von 'TRON' zum nächsten Fehler durchklicken!!

Beschreibung der Funktionen

```
/**************************/
/*      gprintf()         */
/**************************/
```

Die Funktion 'gprintf' entspricht weitgehend der normalen 'printf'-Funktion. Nur daß die Ausgabe nicht auf dem vollen Bildschirm, sondern in kleinen Alertboxen mit je fünf Zeilen zu dreißig ASCII-Zeichen erfolgt.Es ist daher klar, daß Steuerzeichen im Formatstring und in Ausgabestrings nur mit Vorsicht zu verwenden sind.

Ist im Formatstring kein Zeilensprungzeichen '\n' vorhanden, übernimmt gprintf automatisch die Zeilenverteilung und zwar so, daß nach jeder Ausgabe eines Variablenwerts eine neue Zeile angefangen wird. - Ist der Ausgabestring einer Variablen länger als 30 Zeichen, wird nach dem 30. Zeichen automatisch eine neue Zeile angefangen - andernfalls würde die Funktion form_alert Bomben erzeugen.

Hat man aber im Formatstring ein einziges '\n' - Zeichen verwendet, ist man für die Verteilung auf die Zeilen selbst verantwortlich!! gprintf sorgt jetzt nur noch für den Zeilenwechsel nach dem 30. Zeichen. - Diese Betriebsweise von gprintf ermöglicht es zum Beispiel, mehr als eine Variable in eine Zeile zu packen.

Durch Verwendung des Alertbox-Zeilentrenners '|' kann man zu den Zeilentrennern des Automatik-Betriebs weitere zusätzliche Zeilentrenner einfügen.

Insgesamt darf ein gprintf-Befehl 960 Zeichen ausgeben - natürlich verteilt auf mehrere aufeinanderfolgende Alertboxen.

Falls der ABBRUCH - Button angeklickt wurde, gibt geprintf den Wert 0 an die aufrufende Funktion zurück - andernfalls den Wert 2.

```
/**************************/
/*         dump()         */
/**************************/
```

Die dump-Funktion verlangt als Argument eine Speicheradresse (also) long - Format. Sie erzeugt ein Hex-Dump von vier Zeilen zu jeweils 8 Byte ab der angegebenen Speicherstelle. In einer fünften Zeile werden 30 der 32 Bytes als Zeichenkette ausgegeben. Die letzten beiden Zeichen passen leider nicht mehr in die Box! Zeichen mit Zifferncodes unter 31 und oberhalb 166 werden als Punkte ausgegeben.

Durch Anklicken des WEITER-Buttons können jeweils die nächsthöheren 32 Bytes des Speichers ausgegeben werden.

Variable Funktionsparameter

Der Quelltext ist ausführlich kommentiert und selbsterklärend. Eine Wiederholung der Kommentare an dieser Stelle würde keine zusätzliche Information bringen. - Lediglich die Argumentübergabe an gprintf muß näher erklärt werden. Da die Funktion gprintf Argumente unterschiedlichen Typs und in unterschiedlicher Anzahl verarbeiten muß, kann die normale Argumentübergabe hier nicht verwendet werden. gprintf macht sich bei der Argumentübernahme das Verfahren zunutze, mit dem die aufrufende Funktion die Argumente generell übergibt - nämlich über den "Stack". Mit Hilfe der dump-Funktion läßt sich dieses Verfahren leicht demonstieren:

Nehmen wir an, gprintf wird an irgendeiner Stelle des Programms mit den folgenden Argumenten aufgerufen:

```
int  i = 0x1234;
long j = 0x22222222;
static char k[] = "Dies ist ein
String"

gprintf("Var. i = %xVar. j =
l%xVar. k = %s",i,j,k);
```

Das Programm legt nun diese Argumente von rechts beginnend nacheinander auf den Stack: erst die vier Bytes der String-Adresse k, dann die vier Bytes der long-Variablen j , die beiden Bytes von i (int!) und dann die vier Adreßbytes der form_str - Variablen. Zuletzt wird die Rückkehradresse auf den Stack gelegt!! Dabei darf man nicht vergessen, daß der Stack von hohen zu niedrigen Speicheradressen - also nach unten wächst!!

Alertbox 2

```
9E00C: 00 09 C2 76 12 34 22 22
9E014: 22 22 00 09 C1 FC 00 00
9E01C: 00 00 00 00 00 00 00 00
9E024: 00 00 00 00 00 00 00 00
...v.4""""..................
[ABBRUCH] [WEITER] [ OK ]
```

gprintf kennt nun die Anfangsadresse dieses Stacks über die Adresse des ersten Arguments. In der Alertbox 2 ist das Hexdump dargestellt, das sich beim Aufruf von dump(&form_str) ergibt.

Die ersten vier Bytes enthalten die long - Zahl 9C276, nämlich die Adresse eben dieser Variablen 'form_str', danach folgen die Variablen i (1234) und k(22222222) sowie die Adresse des Strings k. (0009C1FC). Alertbox 3 geht der Adresse 9c276 nach und druckt prompt den Formatstring aus:

nächste Seite...

Alertbox 3

```
9C276: 56 61 72 2E 20 69 20 3D
9C27E: 20 25 78 56 61 72 2E 20
9C286: 6A 20 3D 20 25 6C 78 56
9C28E: 61 72 2E 20 6B 20 3D 20
Var. i = %xVar. j = %lxVar. k
[ABBRUCH] [WEITER] [OK]
```

Um diese Variablen aber in der richtigen Weise übernehmen zu können, verfährt gprintf nun folgendermaßen: Zunächst einmal wird der Zeichenpointer 'char *pargu' auf die Anfangsadresse der zweiten Variablen 'argu' gesetzt. Durch die Analyse des Formatstrings erfährt die Funktion, daß es sich bei diesem Argument um den Typ 'int' handelt und übergibt diesen Sachverhalt an das im Programmkopf definierte Funktions-Macro 'arg_wert'. Dieses erhöht nun den Wert des Pointers pargu um die Bytezahl einer int-Zahl: pargu += sizeof(int) (Da es sich um einen charpointer handelt, funktioniert die Addition hier normal wie bei einer long-Zahl). Nachdem der Zeiger nun genau hinter die int-Variable i positioniert ist, wird er durch den Cast (int *) zu einem Zeiger auf ein array von int-Zahlen gemacht und mit (...) [-1] das hinter dem pointer liegende Element angesprochen.

Das gleiche Verfahren wird nun nacheinander mit den übrigen Argumenten durchgeführt. Man sieht aber, daß gprintf und natürlich auch printf selbst ganz schön durcheinander kommt, wenn die Zahl oder der Typ der im Formatstring angegebenen Argumente nicht mit der Zahl oder dem Typ der tatsächlich übergebenen Argumente übereinstimmt. In diesem Fall hat die Übergabe offensichtlich geklappt, denn gprintf gibt die folgende Alertbox aus:

```
Var. i = 1234
Var. j = 22222222
Var. k = Das ist ein String
[ABBRUCH] [TRON] [OK]
```

Einbindung der Funktionen in eigene Programme.

Die Verwendung der Funktionen in eigenen Programmen sollte keine Probleme aufwerfen. Lediglich 'tr' muß als globale Variable im Programm deklariert werden und bei den Programmdefinitionen müssen die Funktionen: arg_wert(pargu, type) und Super(stack) enthalten sein. Ansonsten wird der Funktionsblock irgendwo ins Programm kopiert. Vielleicht kann er sich ja beim nächsten Bombenalarm schon nützlich machen.

Dr. Peter Härtel

```
 1:
 2:  /*************************************************/
 3:  /*************************************************/
 4: /*                                                 */
 5: /*                 BUGALERT.C                      */
 6: /* (Mini-Debugging-Tool für C-Programme unter GEM) */
 7:  /*************************************************/
 8:  /*************************************************/
 9:
10: /*——————————————————*/
11: /*    Programm - Definitionen       */
12: /*__________________________________*/
13:
14:
15: #define arg_wert(pargu,type)
16:         ((type *)(pargu += sizeof(type)))[-1]
            /* Argument fangen */
17: #define Super(stack) gemdos(0x20,stack)
            /* Supervisormodus */
18:
19: /*——————————————————*/
20: /*    Globale Variable              */
21: /*__________________________________*/
22:
23: int     tr;         /* Trace-Schalter */
24:
25: /*——————————————————*/
26: /*  gprintf                         */
27: /*——————————————————*/
28:
29: gprintf(form_str, argu)
                      /* Ersatz für printf in GEM-Pro- */
30: char *form_str; /* grammen. Formatstring wie bei */
31:                 /* 'printf'.                     */
32: {
33: char *pargu = (char *) &argu; /* C-Zeiger auf
                                     Argumente-Stack */
34: char atext[200];   /* Puffer für Alerttext */
35: char text[1000];   /* Puffer für Textausgabe */
36: char *s;           /* Zeiger auf diesen Puffer */
37: char f_str[13];    /* Formatanweisung 12 Zeichen */
38: char *p;           /* Zeiger auf Formatanweisung */
39: char auswahl[22]; /* String für Auswahl-Buttons */
40: int cz = 0,        /* Anzahl Zeichen von sprintf */
41:     zz,            /* Anzahl Zeilen im Text */
42:     vt,            /* Argumenttyp 1=int, 2=long.. */
43:     wahl,          /* Returnwert von form_alert */
44:     autom,         /* automatische Zeilentrennung */
45:     laenge;        /* Länge des Ausgabestrings */
46:
47:
48: s = &text[0];      /* Zeiger auf Puffer setzen */
49: strcpy(atext,"[0]["); /* Anfang Alertbox-strings */
50: strcpy(auswahl,"][ABBRUCH|TRON | OK ]");
51: if(tr)                /* falls tr = wahr,dann TRON - */
52:    {                  /* in TROFF verwandeln */
53:    auswahl[13]='F';
54:    auswahl[14]='F';
55:    }
56: p = form_str;
57:
58: while((*p++) && (*p != '\n')) /* Prüfen, ob '\n'
                                     vorhanden */
59:    ;
60: autom = (*-p) ? 0 : 1;        /* falls vorhanden,
                                     autom=FALSE */
61: /*——————————————————————————*/
62: /*—Formatstring in einen Textstring von max.—*/
63: /*————————950 Zeichen umwandeln————————*/
64: for(laenge = 0;*form_str;)
65:    {
66:    if(*form_str != '%')   /* solange keine
                                 Formatanweisung */
67:       {
68:       laenge++;              /* Zeichen zählen */
69:       *s++ = *form_str++; /* Text in Textpuffer
                                 schreiben */
70:       }
71:    else                /* Variablen - Formatanweisung  */
72:       {
73:       strncpy(f_str,form_str,12); /* 12 Zeichen
                                        Formatanweisung */
74:       p = f_str;   /* wieviel davon sind gültig? */
75:       while((*++p<='9' && *p>='0')
76:                       || *p == '.') /* Feldweite in
                                        Ziffern & Punkt  */
77:          ;          /* falls vorhanden Zeiger++ */
78:       switch(*p)   /* auf gültiges Zeichen prüfen */
79:          {
80:          case 'h': /* h hei_t short-Variable */
81:             p++;   /* nächstes Zeichen ist wichtig */
```

```
82:         case 'c':
83:         case 'd':
84:         case 'x':
85:         case 'o':
86:         case 'u':
87:            vt =1; /* Variablentyp int */
88:            break;
89:         case 'l':
90:            vt =2; /* l hei_t long-Variable */
91:            p++;   /* nächstes Zeichen ist wichtig */
92:            break;
93:         case 's':
94:            vt =3; /* String-Pointer ist gemeint */
95:            break;
96:         case 'f':
97:         case 'e':
98:         case 'g':
99:            vt =4; /* Variable vom Typ double */
100:           break;
101:        case '%': /* Nur Prozentzeichen schreiben */
102:           vt =5;
103:           break;
104:        default:  /* alle and. Zeichen -> error */
105:           form_alert(1,"[3]
              [Formatanweisung nicht korrekt]
                 [ABBRUCH]");
106:           return(1);
107:        }
108:
109:     *++p='\0'; /* Stringende Variablenformat */
110:     form_str += strlen(f_str);
                 /* Pointer auf nächstes Zeichen */
111:
112:
113:     switch(vt) /* Je nach Variablentyp die rich- */
114:        {       /* tige Variable vom Stack holen  */
115:        case 1:
116:          cz=sprintf(s,f_str, arg_wert(pargu,int));
                                                 /* int */
117:          break;
118:        case 2:
119:          cz=sprintf(s,f_str, arg_wert(pargu,long));
                                                /* long */
120:          break;
121:        case 3:
122:          cz=sprintf(s,f_str,
               arg_wert(pargu,char *));   /* pointer */
123:          break;
124:        case 4:
125:         cz=sprintf(s,f_str, arg_wert(pargu,double));
                                              /* double */
126:          break;
127:        case 5:
128:           if(*-s == '|')
129:              {
130:              *s++ = '%';
131:              *s++ = '|';
132:              continue;
133:              }
134:           else
135:              {
136:              *++s='%';
137:              ++s;
138:              continue;
139:              }
140:        }
141:     laenge += cz+1;
142:     if(laenge > 960)
143:        *form_str ='\0'; /* aufhören bevor Puffer
                              voll ist */
144:     s += cz;
145:     if(autom && !(vt==5)) /* falls automatische
                                 Zeilentrennung */
146:        *s++ = '|'; /* neue Zeile für neue
                          Variable! */
147:     }              /* Ende else  */
148:  }                 /* Ende while */
149: if(cz && autom)   /* letztes '|'-Zeichen löschen! */
150:    s-;
151: *s = '\0';         /* String-ende markieren! */
152: /*———Textstring für die Ausgabe———*/
153: /*——in mehreren Alert-Boxen vorbereiten.——*/
154: /*         cz=Zeichenzähler zz=Zeilenzähler        */
155: /*          Zeiger auf Stringanfang setzen          */
156: /*  und bis zum Ende des Textstrings vorarbeiten    */
157:
158: for(cz=1,zz=0,s=&text[0],p=&atext[4];
      *s ;cz++,p++,s++)
159:   {
160:   if((*s<31) || (*s>166)    /* handelt es sich um
                                     Textzeichen? */
161:      ||(*s==91)||(*s==93))  /* [ und ] sind nicht
                                    erlaubt */
162:      *p='?';
163:   else
164:      *p = *s;  /* Zeichen in Ausgabetext kopieren */
165:
166:   if(   (*s == '|' )  /* Zeilentrenner gefunden */
167:      || (*s == '\n')
168:      || (cz == 30   )) /* Zeilenende ohne Trennung
                              erreicht */
169:      {
170:      if(cz == 30)
171:         p++;           /* Zeiger auf nächstes
                              Zeichen */
172:      while(cz++<30)    /* Rest der Zeile mit space
                              füllen */
173:         *p++=' ';
174:      *p = '|';         /* Zeilentrenner in
                              Ausgabetext */
175:      cz=0;             /* Zeichenzähler
                              zurücksetzen */
176:      if(++zz == 5)     /* Zeilenzähler erhöhen */
177:         {              /* falls vierte Zeile, */
178:         *p='\0';       /*  dann Ausgabeende */
179:         strcat(atext,auswahl); /* Ende form_alert-
                                      string anfügen  */
180:
181:         wahl=form_alert(3,atext); /* Alert-box
                                          ausgeben */
182:
183:         if(wahl==2) /* TRON-TROFF Button wurde
                            gewählt */
184:            tr = !tr;
185:         if(wahl==1) /* Abbruchbutton wurde
                            gewählt! */
186:            return(0);
187:         p = &atext[4];
188:         cz = 1;
189:         zz = 0; /* Zeilenzähler zurücksetzen */
190:         }
191:      }
192:   }             /* Ende for - Schleife */
193: if(!zz)          /* falls keine vollständige Zeile */
194:    while(cz++ < 31) /* Rest der Zeile mit space
                           füllen */
195:       *p++=' ';
196: *p = '\0';
197:
198: strcat(atext,auswahl);
199: wahl = (form_alert(3,atext)); /* Alert-box
                                      ausgeben */
200: if(wahl==2)    /* TRON-TROFF Button wurde gewählt */
201:    tr = !tr;   /* Schalter umlegen */
202: return(wahl-1); /* falls ABBRUCH dann Null zurück */
203: }
204:
205: /*——————————————*/
206: /*     dump                          */
207: /*——————————————*/
208:
209: dump(adr)          /* Gibt 32 Bytes ab RAM-Adresse */
210: char *adr;         /* 'adr' als Hex-Zahlen aus */
211: {
212: int i,j;           /* Schleifenzähler */
213: unsigned int byte; /* auszugebendes Byte */
214: int h_n,           /* higher Nibble */
215:     l_n,           /* lower  Nibble */
216:     wahl;          /* Alert-Box-Button */
217: long hexad;        /* Adresse als Zahl */
218: char c;            /* Zeichenwert des Byte*/
219: char atext[200];   /* Ausgabestring */
220: char *s;           /* Zeiger auf atext */
221: char *t;
222: char *p;
223: char *pt;
224: long save_s;
225:
226:
227: strcpy(atext,"[0]["); /* Anfang Alertbox */
228: if((long)adr<0x800)   /* Prüfen, ob Supervisor-Modus
                              nötig */
```

```
229:     if(form_alert(1,
230:     "[3][Die angegebene Adresse liegt|
           im geschützten Speicherbereich|\
231:  Abbrechen ?][JA|NEIN]") == 1)
232:         return(0);
233:
234:  while(1)                    /* bis ABBRUCH gewählt */
235:     {
236:     if((long)adr<0x800)   /* falls nötig,
                                 Supervisormodus->ein */
237:         save_s = Super(0L);
238:
239:     p  = adr;                /* Zeiger auf RAM-Adresse */
240:     s  = &atext[4];          /* Zeiger auf Hex-Ausgabe */
241:     pt = &atext[128];        /* Zeiger auf Textausgabe */
242:
243:     for (i=0;i<4;i++,s++) /* vier Zeilen ausgeben   */
244:         {
245:         t = s+6; /* Adresse rückwärts ausgegeben */
246:         s += 7;
247:         *t--=' ';
248:         *t--=':';
249:         hexad = (long)(adr+8*i);
250:         for(j=0;j<5;j++) /* Hexadresse 5-stellig
                                 ausgeben */
251:             {
252:             byte = (int)hexad & 0xFL; /* nur die letzte
                                        Stelle interessiert*/
253:             *t--=byte+((byte>9) ? 55:48); /* Hex-Zeichen
                                                  ausgeben */
254:             hexad /= 16; /* nächste Stelle ausgeben */
255:             }
256:         for (j=0;j<8;j++) /* Eine Zeile mit 8 Byte
                                  ausgeben */
257:             {
258:             byte = *p++; /* hole byte aus Speicher */
259:                          /* Nur Textzeichen ausgeben */
260:             *pt++=((byte>31 && byte<91) ||
                    (byte>93 && byte<166)) ? byte : '.';
261:             h_n  = byte>>4; /* high-Nibble = h_n */
262:             *s++ = h_n +((h_n > 9) ? 55:48); /* Dez/Hex-
                                                  Umwandlung */
263:             l_n  = byte & 0xF;  /* lower-Nibble = l_n */
264:             *s++ = l_n +((l_n > 9) ? 55:48);
265:             *s++=' ';          /* Leerzeichen nach Byte */
266:             }
267:         *-s='|';              /* Zeilenende markieren */
268:         }
269:     pt -= 2;                  /* Stringende markieren */
270:     *pt='\0';
271:     if((long)adr<0x800)       /* Supervisormodus aus */
272:         Super(save_s);
273:
274:     strcat(atext,"][ABBRUCH|WEITER|  OK  ]");
         /* Buttontext anhängen */
275:
276:     if((wahl = form_alert(3,atext)) != 2)
         /* Alertbox ausgeben */
277:         return(wahl-1); */
278:                            /* falls Wiederholung (2) */
279:     adr += 32;             /* neue Adresse im RAM    */
280:     }
281: }
```

Betr.: Programmier Praxis

Die “Progammier Praxis” ist ein fester Bestandteil der ST Computer geworden. Gedacht ist sie für all diejenigen, die ihren Rechner selbst programmieren, und dazu Tips und Anregungen gebrauchen können. Diese Rubrik kann aber nur dann bestehen, solange viele Leser (Sie eingeschlossen) sich daran beteilligen. Wir fordern Sie deshalb auf, Ihre Ideen in einen Umschlag zu stecken und auf dem schnellsten Weg zu uns zu schicken. Voraussetzungen gibt es praktisch keine: es muß lediglich interessant sein und so dokumentiert, daß auch andere Anwender daraus einen Nutzen ziehen können. Auch bei den Sprachen gibt es keinerlei Einschränkungen: PASCAL, C, BASIC; Assembler, Modula 2, APL, Fortan, LISP, Prolog, PEARL,... und sogar LOGO ist erwünscht.
Veröffentliche Programme werden natürlich angemessen honoriert. Bitte beachten Sie bei der Einsendung: Schicken Sie den Quelltext und das evtl. compilierte Programm, sowie die Dokumentation auf Papier und Diskette. Die Diskette wird zurückerstattet.
Einsendungen direkt an die Redaktion:

“MERLIN” Computer GmbH
ST Computer Redaktion
´Programmier Praxis´
Industriestraße 26
D-6236 Eschborn Tel.: 0 61 96 / 48 18 11

DISK SCHREIB-GESCHÜTZT?

Ihr ST warnt Sie, wenn nicht

Haben Sie nicht auch schon öfter mal eine Diskette aus Versehen gelöscht? Dieses kleine Programm kann Ihnen hier unter die Arme greifen. Denn es wird Sie jedesmal mit einem Ton warnen, wenn eine nicht schreibgeschützte Diskette in das Laufwerk gelegt wird.

Die Bedienung des Programmes ist äußerst einfach. Es kann mit nahezu jedem Assembler bearbeitet werden. Es könnten eventuell Probleme bei den Kommentaren und am Ende des Listings bei den DS.x-Befehlen auftreten. Durch das Weglassen der Kommentare und Ersetzen der DS.x- durch BLK.x-Mnemonics dürften keine Fehler mehr vorkommen.

Nach erfolgreichem Assemblieren kann man das erhaltene Programm aus dem Desktop starten oder man kopiert es in den AUTO-Ordner der Startdiskette. Dort installiert es sich dann selbst.

Nun wird man jedesmal durch einen Ton gewarnt, wenn eine Diskette mit fehlendem Schreibschutz eingelegt wird.

Es muß nur noch beachtet werden, daß das Programm durch einen RESET verloren geht, d.h. es muß erneut installiert werden.

Wie funktioniert das alles ?

Als erstes muß das Einlegen einer Diskette ohne Schreibschutz festgestellt werden. Dank dem TOS hat man es sehr leicht, denn in der Speicherzelle h&9B2 wird dieses Ereignis durch den Wert Null angezeigt. Ansonsten enthält sie den Wert $FF.

Da es leider keine andere Möglichkeit gibt, als die Abfrage in den VBL-Interrupt einzubinden, muß man den Wechsel von $FF auf $00 feststellen. Würde man dies nicht so lösen, hätte man einen Dauerton, solange die Diskette eingelegt ist.

Der Wechsel wird durch das Zwischenspeichern des Wertes aus der Adresse $9B2 erkannt. Der jeweils aktuelle Inhalt dieser Speicherzelle wird mit dem alten gespeicherten Wert verglichen. Tritt hier ein Unterschied auf, fand ein Wechsel statt. Es muß nur noch geprüft werden, ob es sich um die Änderung von $FF auf $00 handelt.

Hat man auf diese Weise erkannt, daß eine Diskette ohne Schreibschutz eingelegt wurde, geht es an den Soundchip. Die Programmierung dieses Chips sieht so aus, daß man ein Register anwählt und dann einen Wert hineinschreibt.

Es wurden folgende Adressen des Soundchips verwendet:

$FF8800 In dieser Speicherzelle kann ein Register angewählt und ebenso ausgelesen werden.

$FF8802 Hier kann man einen Wert in das gewählte Register schreiben.

Will man nun dem Soundchip einen Ton entlocken, muß man als erstes einen Kanal einschalten. Um einen klaren Ton zu erhalten, wird das Rauschen abgestellt. Das zugehörige Register ist hier das siebte. Dort werden die entsprechenden Bits gesetzt oder gelöscht.

Der nächste Schritt ist das setzen der Frequenz für den Ton. Hierzu ist das erste Register zuständig.

Da das Programm im VBL abläuft, kann man sich aus Zeitgründen nicht um die Dauer des Tones kümmern. Doch hier bietet sich die Möglichkeit an, eine Hüllkurve des Soundchips zu benutzen, d.h. der Chip kümmert sich selber um die Dauer. Das hat allerdings den Nachteil, daß keine schönen Klänge verwirklicht werden können. Um die Hüllkurve festzulegen, muß man das 12. und 13. Register verwenden.

Wer sich noch für die Installation der Routine im VBL-Interrupt interessiert kann dies in der Januar '88-Ausgabe auf Seite 84 nachlesen.

Markus Siebler

```
1:  * Computer meldet sich mit einem, Ton wenn
2:  * eine nicht schreibgeschuetzte Diskette in's
3:  * Laufwerk gelegt wird .
4:  *
5:  * (C) Markus Siebler
6:  *
7:
8:  * Programmlaenge veststellen
9:
10:         move.l  $4(a7),a0
11:         move.l  $c(a0),d0
12:         add.l   $14(a0),d0
13:         add.l   $1c(a0),d0
14:         add.l   #$100,d0
15:         move.l  d0,len
16:
17: * Superviser anschalten
18:
19:         clr.l   -(sp)
20:         move.w  #$20,-(sp)
21:         trap    #1
22:         add.l   #6,sp
23:         move.l  d0,save_ssp
24:
25: * Routine in VBL-Liste eintragen
26:
27:         move.l  $456,a1
28:         clr.l   d1
29: loop:
30:         tst.l   $0(a1,d1)
31:         beq     free
32:         addq    #4,d1
33:         jmp     loop
34: free:
35:         tst.l   d1
36:         bne     m3
37:         addq    #4,d1
38:         bra     loop
39: m3:
40:         lea.l   $0(a1,d1),a2
41:         move.l  #vbl,a1
42:         move.l  a1,(a2)
43:
44: * Programm verlassen
45:
46:         jmp     ende
47:
48: * VBL-Routine
49:
50: vbl:
51:         move.l  d0,-(sp)        * D0 zwischenspeichern
52:
53:         move.b  $9b2,d0   * $9b2 enthaelt WP-Status
54:         cmp.b   ws,d0     * hat sich was geaendert ?
55:         beq     quit      * Nein -> quit
56:         cmp.b   #$ff,d0   * Ja -> wurde Disk eingelegt ?
57:         beq     quit      * Nein -> quit
58:
59:         move.b  #7,$ff8800 * Ja -> Soundchip Reg.7
                                  anwaehlen
60:         move.b  $ff8800,d0 * in D0 speichern
61:         and.b   #$fe,d0    * Kanal 1 Ton ein
62:         or.b    #$8,d0     * Kanal 1 Rauschen aus
63:         move.b  d0,$ff8802 * D0 wieder in Reg.7
                                 schreiben
64:
65:         move.b  #1,$ff8800 * Frequenz des Tones
                                 einstellen
66:         move.b  #3,$ff8802
67:
68:         move.b  #8,$ff8800   * Lautstaerke einstellen
69:         move.b  #$10,$ff8802 * Huellkurve waehlen
70:
71:         move.b  #12,$ff8800  * Frequenz der Huellkurve
72:         move.b  #7,$ff8802
73:
74:         move.b  #13,$ff8800  * Huellkurve waehlen
75:         move.b  #4,$ff8802
76:
77: quit:
78:         move.b  $9b2,ws      * WP-Status in <ws>
                                   speichern
79:         move.l  (sp)+,d0     * D0 wieder in alten
                                   Zustand
80:         rts                  * weiter geht's im VBL
81:
82: * Supervisor aus
83:
84: ende:
85:         move.l  save_ssp,-(sp)
86:         move.w  #$20,-(sp)
87:         trap    #1
88:         add.l   #6,sp
89:
90:         * Programm verlassen und resident halten
91:
92:         move.w  #0,-(sp)
93:         move.l  len,-(sp)
94:         move.w  #$31,-(sp)
95:         trap    #1
96:
97:
98: len:
99:         ds.l    1        * Programmlaenge
100: save_ssp:
101:        ds.l    1        * USER-Stack-Pointer
102: ws:
103:        ds.b    1        * WP-Status
104: cnt:
105:   dc.b    -1            * Tondauer
```

```
10 ' W_PROTEC.PRG Lader (Zeilennummern nicht notwendig)
11 Filename$= "W_PROTEC.PRG"
12 OPEN "O",1,Filename$
13 REPEAT
14   C=C+1
15   READ Wert
16   IF Wert=-1 THEN
17     END
18   ENDIF
19   IF (C MOD 11)=0 THEN
20     IF Wert<>(Summe AND 255) THEN
21       PRINT"Fehler in Datazeile "C/11
22       END
23     ENDIF
24     Summe=0
26   ELSE
27     PRINT #1, CHR$(Wert);
28     Summe=Summe+(Wert+C MOD 11)^2
29   ENDIF
30 UNTIL Wert=-1
31 CLOSE(1)
100 DATA 96,26,0,0,1,22,0,0,0,0,68
101 DATA 0,0,0,0,0,0,0,0,0,0,129
102 DATA 0,0,0,0,0,0,0,0,32,111,142
103 DATA 0,4,32,40,0,12,208,168,0,20,65
104 DATA 208,168,0,28,6,128,0,0,1,0,100
105 DATA 35,192,0,0,1,12,66,167,63,60,107
106 DATA 0,32,78,65,223,252,0,0,0,6,149
107 DATA 35,192,0,0,1,16,34,121,0,0,124
108 DATA 4,86,66,129,74,177,16,0,103,0,242
109 DATA 0,10,88,65,78,249,0,0,0,56,51
110 DATA 74,129,102,0,0,8,88,65,96,0,135
111 DATA 255,230,69,241,16,0,34,124,0,0,44
112 DATA 0,102,36,137,78,249,0,0,0,234,23
113 DATA 47,0,16,57,0,0,9,178,176,57,145
114 DATA 0,0,1,20,103,0,0,102,176,60,83
115 DATA 0,255,103,0,0,94,19,252,0,7,69
116 DATA 0,255,136,0,16,57,0,255,136,0,172
117 DATA 2,0,0,254,0,0,0,8,19,192,252
118 DATA 0,255,136,2,19,252,0,1,0,255,7
119 DATA 136,0,19,252,0,3,0,255,136,2,182
120 DATA 19,252,0,8,0,255,136,0,19,252,180
121 DATA 0,16,0,255,136,2,19,252,0,12,41
122 DATA 0,255,136,0,19,252,0,7,0,255,131
123 DATA 136,2,19,252,0,13,0,255,136,0,174
124 DATA 19,252,0,4,0,255,136,2,19,249,109
125 DATA 0,0,9,178,0,0,1,20,32,31,168
126 DATA 78,117,47,57,0,0,1,16,63,60,32
127 DATA 0,32,78,65,223,252,0,0,0,6,149
128 DATA 63,60,0,0,47,57,0,0,1,12,231
129 DATA 63,60,0,49,78,65,0,0,0,49,251
130 DATA 0,0,0,49,0,255,0,0,0,24,127
131 DATA 20,24,22,8,14,114,10,22,0,0,97
9999 Data -1
```

Das Assembler-Listing als BASIC-Datas

REISE INS CHAOS

Fractals

Wer hat sie noch nicht gehört, die Begriffe wie Mandelbrot-Menge, Julia-Menge oder Apfelmännchen? Hinter diesen Begriffen steckt der Reiz des Unendlichen, die Faszination skurriler Gebilde. Wohl jeder, der Fractals erblickt hat, ist von ihnen beeindruckt.

Legen Sie den Artikel nicht gleich weg, wenn Sie jetzt zuerst etwas über Mathematik lesen. Die mathematische Grundlage für unsere Fractals ist überaus einfach.

Das Reizvolle an einem Fractal ist seine Eigenschaft, fraktal zu sein. Fraktal heißt ein Gebilde, um es trocken auszudrücken, wenn es selbstähnlich ist, und wenn der Rand unendlich zerklüftet ist. Mit anderen Worten heißt dieses, daß man jedes Gebilde oder Teilgebilde an beliebigen Stellen unendlich oft wiederfindet, jedesmal noch schöner und reizvoller. An keinem Stück des Randes wird man ein glattes Stück finden, immer ergeben sich neue Randformen. Es sieht chaotisch und geordnet zugleich aus. Gerade diese beiden Eigenschaften machen der Reiz der Fractals aus!

Hinter den Begriffen Mandelbrot-Menge, Julia-Menge und Apfelmännchen steckt ein und dieselbe Formel. Fangen wir mit der Geschichte an. Die Mandelbrot-Menge wurde nach Benoit B. Mandelbrot, einem aus Polen stammenden Mathematiker, benannt. Gefunden wurde sie erst im Jahre 1980. Seitdem wird ununterbrochen nach weiteren Ablegern gesucht, hier bei uns etwa von einer Bremer Forschungsgruppe. Diese Forschungsgruppe gab den Mandelbrot-Mengen einen Spitznamen: Apfelmännchen. Der Rand der Mandelbrot-Menge hat wieder einen besonderen Namen: Julia-Menge. Die Julia-Mengen, die ihren Namen dem französischen Mathematiker Gaston Julia verdanken, haben einen zum Apfelmännchen analogen Aufbau, mit einer klitzekleinen Veränderung. Zu dieser Veränderung kommen wir später.

Ein Apfelmännchen ist ein ebenes Gebilde. Die Ebene, auf der es entsteht, heißt Zahlenebene oder komplexe Zahlenebene. Im Gegensatz zu den schon meist aus der Schule bekannten reellen Zahlen, die auf einer Zahlengeraden dargestellt werden, werden diese Zahlen in der Ebene dargestellt.

Neben der waagerechten Komponente, dem Realteil, hat eine komplexe Zahl noch eine senkrechte Komponente, den Imaginärteil. Um die Zahl leichter darstellen zu können, teilt man sie in die beiden Komponenten auf: $z = (a,b)$. Die komplexe Zahl z hat den Realteil a und den Imaginärteil b. Diese Darstellung ist ähnlich einer gewöhnlichen Ebenendarstellung. Es gibt aber noch eine zweite Schreibweise: $z = a+ib$. Die Zahl i hat die ganz besondere Eigenschaft $i^2 = -1$, sie ist also die Quadratwurzel aus -1! Im Bild 1 ist eine graphische Darstellung der komplexen Zahlen zu sehen.

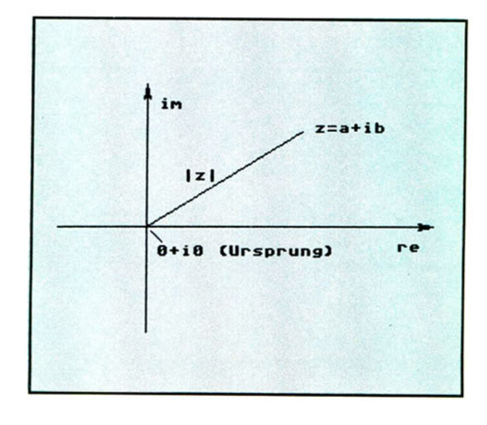


Bild 1: Grafische Darstellung komplexer Zahlen

Jetzt, wo wir die komplexen Zahlen kennengelernt haben, merken wir schon, daß man wohl gewohnt addieren und subtrahieren, nicht jedoch multiplizieren und dividieren kann.

Es wird eine neue Multiplikation eingeführt:

$$yz = ac-bd+i(ad+bc),$$

wobei $y = a+ib$, $z = c+id$ komplexe Zahlen sind.

Ein Beispiel:

$(5+3i)*(-3+4i) = 5*(-3)-3*4+i*(5*4+3*(-3)) = -27+11i$

Eine Division direkt ist nicht möglich. Dazu wird das komplex Konjugierte der komplexen Zahl benötigt. Da wir die Division zweier komplexer Zahlen für unsere Betrachtungen nicht brauchen, wird hier nicht weiter darauf eingegangen.

Der Betrag einer komplexen Zahl $z=a+ib$ berechnet sich folgendermaßen: $|z| = (a^2+b^2)^{0.5}$. Der Betrag ist anschaulich gesehen die Länge der Strecke vom Ebenen-Ursprung zur Zahl $a+ib$.

Bevor wir nun wieder auf die Apfelmännchen zurückkommen, brauchen wir noch ein wenig Mathematik. Eine Folge von Zahlen entsteht mit Hilfe einer bestimmten Vorschrift, einer Iterationsvorschrift. Hierbei können Zahlen als Folgenglieder unter Benutzung einer festen Vorschrift entstehen, die von nichts weiter abhängt, als von der zugrundegelegten Zahl. Die Folge der Stammbrüche (Berechnungsformel 1/n, n ist eine natürliche Zahl, also 1, 2, 3, ...) ist eine solche Folge. Es entstehen hier die Zahlen 1, 1/2, 1/3, ... Bei solchen Folgen kann man relativ leicht vorhersagen, ob sie sich einer festen Zahl beliebig nähern, man spricht von "konvergieren", oder eben nicht, was mit "divergieren"

bezeichnet wird.

Eine andere Art von Folgen entsteht, wenn vorige Folgenglieder benutzt werden, um ein neues zu bestimmen. Diese Folgen heißen rekursive Folgen. Und gerade dieser Typ von Folgen liegt unseren Apfelmännchen zugrunde. Es wird eine Zahl vorgegeben und die nächste wird aus der vorherigen berechnet, indem man sie quadriert und eine Konstante addiert:

$$z_{n+1}=z_n^2+c$$

Das einzige Problem hierbei ist, daß unsere Zahlen komplex sind und wir die komplexe Multiplikation und Addition bemühen müssen. Nehmen wir einmal an, zn sei gleich $x+iy$ und c sei $a+ib$. Dann berechnet sich der Realteil des neuen Folgengliedes als $x*x-y*y+a$ und der Imaginärteil als $2*x*y+b$. Diese Formel werden wir benutzen, um die Apfelmännchen zu berechnen.

An dieser Stelle ist der mathematische Exkurs beendet, wir gehen zur Umsetzung in ein Programm über. Es soll hier allerdings kein vollständiges Programm dargelegt werden, sondern nur einzelne Routinen, die man selbst zusammenfassen kann.

Welche komplexen Zahlen sind überhaupt passend? Grob gesagt, alle um den Ursprung in der komplexen Zahlenebene. Genauer ausgedrückt sollte der Realteil und der Imaginärteil jeweils etwa zwischen -2 und 2 liegen. Neben den Koordinaten aus der komplexen Zahlenebene werden natürlich auch noch Bildschirmkoordinaten gebraucht. Diese Koordinaten beziehen sich auf unsere Pixel. Gleichzeitig muß man also mit zwei verschiedenen Koordinatenpaaren rechnen.

Um es einmal graphisch zu veranschaulichen, betrachten wir Bild 2, wo ein kleines Übertragungsbeispiel zu erkennen ist.

Die Beispiele sind übrigens in GFA-Basic geschrieben, allerdings sind die Routinen so allgemein gehalten, daß man auch jede andere Programmiersprache verwenden kann. Geschwindigkeit ist jedoch Trumpf, da für ein Bild viele tausend Berechnungen durchgeführt werden müssen, was bei ausreichender Rechengenauigkeit mitunter Stunden dauern kann.

Die in eine Prozedur umgesetzte Koordinatenberechnung ist im Listing 1 zu finden.

Der Prozedur FRACTAL werden folgende Werte übergeben:

x% : die x-Koordinate der oberen linken Bildecke auf dem Bildschirm
y% : die y-Koordinate dazu
w%: die Breite des Bildes auf dem Bildschirm
h%: die Höhe dazu
xk: die x-Koordinate der oberen linken Ecke eines Ausschnittes auf der komplexen Zahlenebene
yk: die dazugehörige y-Koordinate
wk: die Breite des Ausschnittes auf der komplexen Zahlenebene
hk : die Höhe dazu

Max_iter% : Obergrenze für die Iteration
A,B: Konstanten

V_... : variierende Werte

Die Iteration kann natürlich nicht unbeschränkt ausgeführt werden, daher beschränken wir uns auf dem Rechner auf höchstens Max_iter% Werte und schätzen dann die Konvergenz ab: Wird Max_iter% erreicht, ohne daß die Folgenglieder eine bestimmte Schranke überschreiten, nehmen wir an, daß die Folge mit den vorgegebenen Startwerten konvergent ist, sonst eben divergent.

Da bei Divergenz die magische Obergrenze Max_iter% nicht erreicht wird, findet diese Abbruchzahl auch noch Verwendung. Das sehen wir allerdings erst später.

Die beiden angesprochenen Prozeduren MANDELBROT und JULIA enthalten die eigentliche Konvergenzbestimmung. Beide Prozeduren sind recht ähnlich, und doch unterscheiden sie sich.

Für die altbekannte Mandelbrotmenge wird der z-Wert konstant gehalten. Hier bringen zwar unterschiedliche Konstanten auch unterschiedliche Graphiken, der Unterschied ist aber nicht so berauschend wie bei den Julia-Mengen. Eine Veränderung der Konstanten lohnt sich also nur für die Julia-Mengen. Konstant gehalten werden natürlich nur die Startwerte!

Die Iteration, getrennt nach Real- und Imaginärteil, befindet sich in der While-Schleife.

Die Prozeduren MANDELBROT und JULIA sind in den Listing 2 und 3 aufgeführt.

Die Variablen, die mit Z_ beginnen, sind die z-Werte aus der Iterationsvorschrift, die mit C_ die c-Werte. "re" steht für Real-Teil und "im" für Imaginärteil.

Der Grundalgorithmus läßt sich in dem in Bild 3 zu sehenden Struktogramm darstellen.

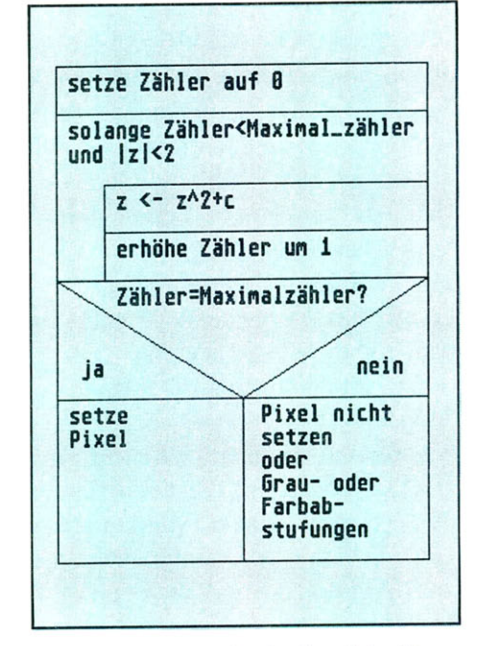


Bild 3: Struktogramm für den Grundalgorithmus der Prozedur MANDELBROT und JULIA

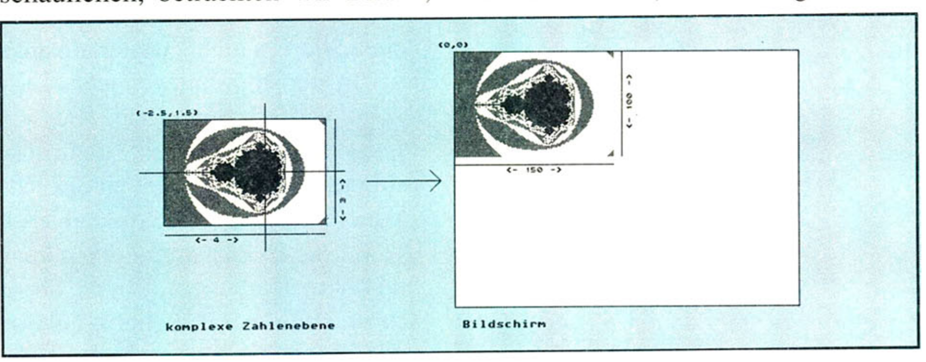


Bild 2: Ein kleines Überraschungsbeispiel

Der jeweils resultierende Wert für Zähler ist für die Pixelausgabe wichtig. Eine einfache Umsetzung könnte so erfolgen:

Setze Pixel,
falls Zaehler%>=Max_iter%,
sonst nicht.

Wer allerdings eine schönere Ausgabe wünscht, der bezieht noch die Punkte der Divergenz mit ein. Erst durch diese Punkte ergibt sich ein (fast) vollendetes Formen- und Farbenspiel. Beispielsweise können graue Bereiche eingeführt werden, wie es in dem Ausgabebeispiel der Fall ist. Ebenso könnte man aber auch je nach Wert die Pixel in einer anderen Farbe setzen. Es gilt aber nicht unbedingt, daß viele Farben auch ein besonders tolles Bild ergeben. Weniger ist oft mehr!

Eine Beispielausgabe ist im Listing 4 zu betrachten. Die vier vorgestellten Prozeduren kann man auch gut zu einer relativ kompakten zusammenfassen, was auch der Geschwindigkeit zugute kommt. Eine Zusammenfassung für die Mandelbrot-Mengen ist im Listing 5 zu erkennen. Dieses Listing enthält im Gegensatz zu den anderen ein lauffähiges Programm!

Bei welchen Koordinaten das "Grund"-Apfelmännchen zu finden ist, wurde schon angesprochen. Besonders schöne Ableger gibt's beispielsweise an den Stellen, wo zwei größere Teile des Apfelmännchens zusammenliegen. Auch kleine Flecke, die recht unscheinbar irgendwo liegen, sollte man nicht verachten, denn gerade dort könnte sich ja eine Selbstabbildung der Mandelbrot-Menge befinden.

Die kompaktere Fractal-Routine aus Listing 5 ist zum "Zoomen" geeignet, d.h. man kann sie gut benutzen, um Ausschnitte aus der Mandelbrot- (oder Julia-) Menge zu vergrößern. Die Vergrößerungsmöglichkeit wird ja dringend benötigt, da unsere Monitore und auch unser Auge nur ein begrenztes Auflösungsvermögen haben. Den Rand zoomen wir uns heran, damit er klar genug wird. Als Koordinaten auf der reellen und der imaginären Achse beginnen wir mit etwa -2 und 2, also einem Bereich von 4. Bei starken Vergrößerungen kann es aber leicht passieren, daß man sich nur noch im Bereich von 0.01 oder noch weniger bewegt. Wenn man Pech hat, verliert man den Rand aus den Augen. Daher sollte man sich die Koordinaten der Bilder immer notieren (Koordinaten auf der reellen und der imaginären Achse, übergebene Konstanten und natürlich den Wert für die maximale Iterationsanzahl).

Die Koordinaten könnten beispielsweise für die linke obere Ecke 0.7453444,0.1130534, für die Breite 0.0005516 und die Höhe 0.0000994 lauten. Die Konstanten sind einfach 0, gerechnet wird bis 400! Bei dieser Rechnung sollte man schon etwas Zeit mitbringen. Meine Empfehlung: Rechner abends anschalten und Programm starten und über Nacht laufen lassen.

Für Julia-Mengen wählt man ähnliche Koordinaten wie für die Mandelbrot-Mengen. Zusammenhängende Julia-Mengen erhält man, wenn man für den konstanten Wert c Werte aus dem Inneren der Mandelbrot-Menge wählt.

Wie weit soll die Iteration fortgeführt werden, um eine Aussage über die Konvergenz zu wagen? Diese Frage ist garnicht so leicht zu beantworten. Je kleiner der Ausschnitt ist, desto größer sollte man Max_iter% wählen. Das können ohne weiteres Werte von 100 und mehr sein. Für gröbere Ausschnitte kann man notfalls auch bis etwa 20 heruntergehen, wodurch aber der fraktale Rand schon leiden könnte.

Ist nun schon Schluß? Nein, jetzt geht's erst richtig los. Fractals ergeben sich nicht nur mit der Iterationsvorschrift $z_{n+1} = z_n^2+c$. Wie wäre es mit $z_{n+1} = z_n^3+c$? Oder eine etwas kompliziertere Iteration $z_{n+1}=(z_n^2+1)/z_n(z_n^2-1)$? Eine Grenze fürs Experimentieren ist nicht gesetzt.

Nach diesem kleinen Ausflug in das Land der Fractals sind auch sicher Sie so begeistert, daß Sie sich Ihren Rechner schnappen und sich ein eigenes Fractal-Programm schreiben. Oder lesen Sie den Artikel schon nicht mehr und sitzen bereits am Rechner?

Dietmar Rabich

Literatur:

[1] Spektrum der Wissenschaft 2/88, Computer-Kurzweil, A. K. Dewdney

[2] The Beauty of Fractals, H.-O. Peitgen/ P. H. Richter, Springer-Verlag

[3] Computergrafische Experimente mit Pascal, K.-H. Becker/ M. Dörfler, Vieweg-Verlag

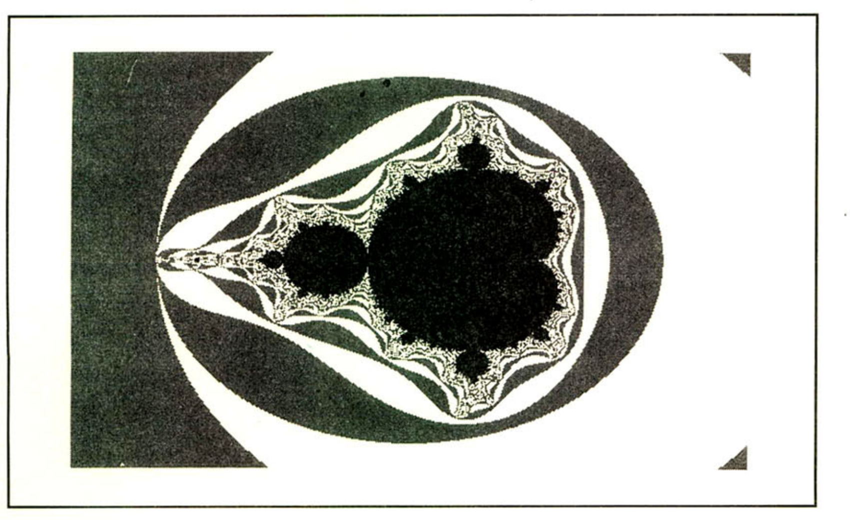


Bild 4: Beispiel für ein Fractal

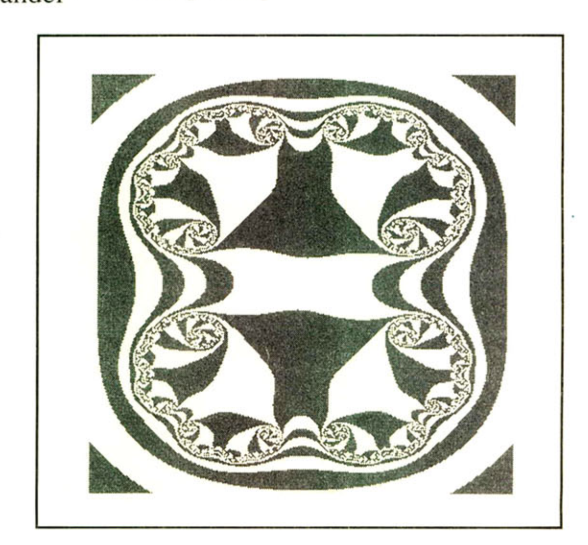


Bild 5: Beispiel für ein Julia-Menge

```
 1:  ' Fractal: Koordinatenberechnung
 2:  Procedure Fractal(X%,Y%,W%,H%,Xk,Yk,Wk,Hk,Max_iter%,
                                                    A,B)
 3:    Local V_re_schritt,V_im_schritt ! Schrittweiten für
 4:    Local I%,J%                     ! Pixelkoordinaten
 5:    '
 6:    V_re_schritt=Wk/W%     ! Schrittweite horizontal
 7:    V_im_schritt=-Hk/H%    ! Schrittweite vertikal
 8:    '
 9:    V_im=Yk                ! Startwert vertikal/oben
10:    For J%=Y% To Y%+H%
11:      V_re=Xk              ! Startwert horizontal/links
12:      For I%=X% To X%+W%
13:        Gosub Mandelbrot                  ! Iteration
14:        Gosub Plot_pixel(I%,J%,Zaehler%)  ! Ausgabe
15:        Add V_re,V_re_schritt
16:      Next I%
17:      Add V_im,V_im_schritt
18:    Next J%
19:  Return
```

Listing 1: Procedure Fractal

```
 1:  ' Fractal: Iteration Mandelbrot-Menge
 2:  Procedure Mandelbrot
 3:    Local Z_re_h
 4:    Local Z_re_m,Z_im_m
 5:    Local C_re_m,C_im_m
 6:    '
 7:    Z_re_m=A              ! z-Werte konstant
 8:    Z_im_m=B
 9:    C_re_m=V_re           ! c-Werte variabel
10:    C_im_m=V_im
11:    '
12:    Zaehler%=0
13:    While Zaehler%<Max_iter% And
           Z_re_m*Z_re_m+Z_im_m*Z_im_m<4
14:      Z_re_h=Z_re_m
15:      Z_re_m=Z_re_m*Z_re_m-Z_im_m*Z_im_m+C_re_m
16:      Z_im_m=2*Z_re_h*Z_im_m+C_im_m
17:      Inc Zaehler%
18:    Wend
19:  Return
```

Listing 2: Iteration Mandelbrot-Menge

```
 1:  ' Fractal: Iteration Julia-Menge
 2:  Procedure Julia
 3:    Local Z_re_h
 4:    Local Z_re_m,Z_im_m
 5:    Local C_re_m,C_im_m
 6:    '
 7:    Z_re_m=V_re              ! z-Werte variabel
 8:    Z_im_m=V_im
 9:    C_re_m=A                 ! c-Werte konstant
10:    C_im_m=B
11:    '
12:    Zaehler%=0
13:    While Zaehler%<Max_iter% And
           Z_re_m*Z_re_m+Z_im_m*Z_im_m<4
14:      Z_re_h=Z_re_m
15:      Z_re_m=Z_re_m*Z_re_m-Z_im_m*Z_im_m+C_re_m
16:      Z_im_m=2*Z_re_h*Z_im_m+C_im_m
17:      Inc Zaehler%
18:    Wend
19:  Return
```

Listing 3: Iteration Julia-Menge

```
 1:  ' Fractal: Pixelausgabe
 2:  Procedure Plot_pixel(Xp%,Yp%,It%)
 3:    ' Konvergenzgebiet (schwarz/wei_):
 4:    '  If It%>=Max_iter%
 5:    '   Plot Xp%,Yp%
 6:    '  Endif
 7:    ' Konvergenzgebiet (schwarz/grau/wei_):
 8:    If It%>=Max_iter%
 9:      Plot Xp%,Yp%
10:    Else
11:      If (It%<=Max_iter% Div 2) And (It% Mod 2=1) And
            ((Xp%+Yp%) Mod 2=1)
12:        Plot Xp%,Yp%
13:      Endif
14:    Endif
15:  Return
```

Listing 4: Pixelausgabe

```
 1:  ' Kompakter Fraktal-Generator
 2:  ' (Entw. mit GFA-Basic.)
 3:  ' D. Rabich, 2.2.1988
 4:  '
 5:  Gosub Fractal(0,0,639,399,-2.5,1.5,4,3,75,0,0)
                                 ! zum Fractal-Generator
 6:  Open "O",#1,"FRACTAL.DOO"     ! Bildausgabe
 7:  Get 0,0,639,399,Bild$
 8:  Print #1,Mid$(Bild$,7);
 9:  Close #1
10:  End
11:  '
12:  ' Fractal-Generator
13:  Procedure Fractal(X%,Y%,W%,H%,Xk,Yk,Wk,Hk,
       Max_iter%,A,B)
14:    Local V_re_schritt,V_im_schritt ! Schrittweiten
15:    Local I%,J%               ! für Pixelkoordinaten
16:    Local Z_re_h              ! für Iteration
17:    Local Z_re_m,Z_im_m
18:    V_re_schritt=Wk/W%        ! Schrittweite horizontal
19:    V_im_schritt=-Hk/H%       ! Schrittweite vertikal
20:    V_im=Yk                   ! Startwert vertikal/oben
21:    For J%=Y% To Y%+H%
22:      V_re=Xk                 ! Startwert
                                   horizontal/links
23:      For I%=X% To X%+W%
24:        Z_re_m=A              ! Iteration Mandelbrot
25:        Z_im_m=B              ! z-Werte konstant
26:        Zaehler%=0
27:        While Zaehler%<Max_iter% And
             Z_re_m*Z_re_m+Z_im_m*Z_im_m<4
28:          Z_re_h=Z_re_m
29:          Z_re_m=2_re_m*Z_re_m-Z_im_m*Z_im_m+V_re
30:          Z_im_m=2*Z_re_h*Z_im_m+V_im
31:          Inc Zaehler%
32:        Wend
33:        ' Die folgende Farbbestimmung könnte man auch
34:        ' in einer Zeile zusammenfassen, aber das würde
           geringfügig komplizierter.
35:        If Zaehler%>=Max_iter%
36:          Color 1
37:        Else
38:          Color -((Zaehler%<=Max_iter% Div 2) And
               (Zaehler% Mod 2=1) And ((I%+J%) Mod 2=1))
39:        Endif
40:        ' Ohne Grauabstufung:
           Color -(Zaehler%>=Max_iter%)
41:        ' Für Farbe (Bsp.: 8 Farben) :
42:        '   If Zaehler%>=Max_iter%
43:        '     Color=0
44:        '   Else
45:        '     Color (Zaehler% Mod 7)+1
46:        '   Endif
47:        Plot I%,J%                       ! Ausgabe
48:        Add V_re,V_re_schritt
49:      Next I%
50:      Add V_im,V_im_schritt
51:    Next J%
52:  Return
53:  ' Ende
```

Listing 5: Der komplette Fractal-Generator

UND SIE DREHT SICH DOCH...



Die lange totgeglaubten Routinen appl_trecord und appl_tplay führten auf den 'alten' STs zu den merkwürdigsten Ergebnissen. Unglaublich, aber wahr: Die Routinen funktionieren im neuen MEGA ST tatsächlich, zwar nicht ganz so, wie in den Dokumentationen beschrieben, aber das ist ja nebensächlich...

Zunächst wollen wir uns den Sinn der Routinen klarmachen. Appl_trecord und Appl_tplay sind Routinen des AES, die dazu dienen, die Ereignisse des GEM aufzuzeichnen, um sie später wieder 'abspielen' zu können. Zu diesem Zweck werden vier verschiedene Ereignisse unterschieden:

0) Timer-Ereignis:
Speichert die Zeit ab, die zwischen zwei anderen Ereignissen vergangen ist.

1) Maustasten-Ereignis:
Zeichnet auf, ob eine Maustaste gedrückt worden ist.

2) Maus-Ereignis:
Wird die Maus bewegt, werden die absoluten X- und Y-Koordinaten der Maus abgespeichert.

3) Tastatur-Ereignis:
Ist eine Taste gedrückt worden, wird der Tastaturcode des Zeichens abgespeichert.

Zum Aufzeichnen der Ereignisse wird die im folgenden beschriebene Routine APPL_TRECORD verwendet:

ap_trreturn = appl_trecord(buffer, evnt_num);

buffer:
Speicherbereich, in den die einzelnen Ereignisse geschrieben werden sollen (benötigte Bytes: = Anzahl der Ereignisse x 8 -> siehe unten).

evnt_num:
Anzahl der Ereignisse, die aufgezeichnet werden sollen.

Um zu wissen, wieviel Speicherplatz wir für unsere Ereignisse reservieren müssen, schauen wir uns das Format an, in dem die Ereignisse von AES abgespeichert werden. Beim Erarbeiten der Parameter der Routine stellte sich leider heraus, daß die Originaldokumentation mal wieder nicht mit der Praxis übereinstimmt.

Format eines Ereignisses:
1. LONG-Word: Dieses LONG-WORD (und nicht wie in der Originaldokumentation beschrieben: INTEGER-WORD) enthält das Ereignis. Die Ereignisse sind durchnumeriert, wobei

```
          Timer den Wert 0,
      Maustasten den Wert 1,
           Maus den Wert 2
und die Tastatur den Wert 3
```

besitzen. Warum von den Programmierern der Routine dafür ein LONG-WORD reserviert worden ist, ist mir eine Rätsel. Ein Byte (!) hätte vollkommen gelangt. Auf diese Weise werden 3 (in Worten: drei) Bytes, die den Wert 0 enthalten, zusätzlich speicherverschwendend zu jedem Ereignis abgespeichert.
2. LONG-Word: Diese vier Bytes enthalten abhängig von dem Ereignis ihren Eintrag (Bei den Beispielen wird immer ein Wort dargestellt):

Timer:
Es wird im LONG-WORD die Anzahl der vergangenen 1/10-Sekunden abgespeichert. Dabei ist die Zeit maßgebend, die zwischen zwei Ereignissen vergangen ist.
Beispiel: 0 0 0 20 - Dieses Ereignis stellt eine Pause von 32 (20 ist hexadezimal) Millisekunden dar.

Maustasten:
Hier wird der Status der Maustasten vermerkt. Die Anzahl der Tastendrücke wird in den ersten zwei Worten und der Status (0 bedeutet Maustaste nicht gedrückt, 1 bedeutet Maustaste gedrückt) in den beiden letzten Worten abgespeichert.
Beispiel: 0 1 2 1 - Die linke Maustaste (1) wurde zweimal (2) gedrückt.

Maus:
Die beiden Wörter dienen dem Abspeichern der absoluten X- und Y-Koordinate der Maus. Die X-Koordinate ist im ersten Wort und die Y-Koordinate im zweiten Wort zu finden.
Beispiel: 0 2 30 A0 - Die Maus ist zur Koordinate (48/160) verschoben worden.

An der Auflistung erkennt man, daß man pro Ereignis acht Bytes reservieren muß. Um eine genaue Mausführung zu ermöglichen, wird natürlich jede Koordinatenverschiebung bei appl_trecord einzeln aufgezeichnet. Wird beispielsweise die Maus von der Koordinate (10,100) auf die Koordinate (110,100) langsam verschoben, so bedeutet dies 100 Ereignisse und damit 800 Bytes - Es lebe der MEGA-ST...Folgerung: Das Aufzeichnen von Daten, besonders wenn die Maus verwendet wird, ist äußerst speicherintensiv.

geht weiter...

Unser Recorder braucht eine WIEDERGABE-Taste

Zur Wiedergabe der Ereignisse wird die Routine appl_tplay des AES verwendet, die wie folgt definiert ist:

ap_tpreturn = appl_tplay(buffer, evnt_num, scale)

buffer:

Enthält wie bei appl_trecord die Adresse des Speicherbereichs, in dem die Ereignisse gespeichert sind. Der Puffer muß mindestens achtmal so groß sein wie evnt_num.

evnt_num:

Dieser Parameter gibt die Anzahl der Ereignisse an, die abgespielt werden sollen.

scale:

Bei scale handelt es sich um einen Skalierungsfaktor, durch den die Geschwindigkeit angegeben wird (Auf keinen Fall die Zahl Null verwenden!!!)

Ein Wort noch zu dem Parameter scale und der Zeitsteuerung. Entgegen der Ausführung wird der Zeiteintrag nicht (!) in tausendstel Sekunden sondern nur in zehntel Sekunden durchgeführt- dies ist eine wichtige Tatsache, denn es ist ja wohl ein relativ großer Unterschied, ob der Rechner eine Pause von einer oder von hundert Sekunden einlegt, nicht ?! In der Originaldokumentation findet man weiterhin unter dem Stichwort scale, daß der Wert 100 normale, der Wert 50 halbe und der Wert 200 doppelte Geschwindigkeit bedeutet. Leider ist auch dies falsch, aber keine Angst, das Rätsel ist gelöst. Die Routine teilt schlichtweg alle Zeitwerte durch den angegebenen Skalierungsfaktor. Was folgt daraus ? Keine Zeitsteuerung in der Ereignisliste, also auch keine Skalierungsmöglichkeit ! Weiterhin ist es nicht sinnvoll, bei Zeitwerten die bespielsweise kleiner 10 sind, einen größeren Skalierungsfaktor als 10 zu verwenden, denn kleiner als 1 kann die Division nicht werden. Übrigens erklärt diese Vorgehensweise auch, warum appl_tplay bei einem Skalierungsfaktor von 0 abstürzt. Bei diesem Skalierungsfaktor wird der Ausnahmevektor "Division durch Null" ausgeführt, was bedeutet, daß der 'intelligente' Programmierer vergessen hat, die Division durch Null an der Skalierungsstelle abzufangen - nobody's perfect. Beispiel 4 zeigt die verschiedenen Skalierungsmöglichkeiten.

Resümee

Alles in allem kann man behaupten, daß die Routinen ganz nützlich sein können, wenn man die richtigen Parameter verwendet. Ich glaube aber, daß ein Aufzeichnen weniger sinnvoll ist als das Erstellen eigener Ereignislisten, denn das Aufzeichnen verbraucht relativ viel Speicherplatz. Erstellt man eigene Listen, kann man gezielt Pausen setzen und auch mal ein paar Mauskoordinaten überspringen, um Speicherplatz zu sparen. Um Ihnen noch eine Idee mit auf den Weg zu geben: Durch ein entsprechendes Programm, das es ermöglicht, hilfreiche Listen zu erstellen, könnte über appl_tplay ein Demonstrationsprogramm generiert werden, daß zum Beispiel eigene Funktionen auf dem Desktop oder in anderen Programmen durchführt.

Noch ein klitzekleiner Tip am Schluß: Appl_tplay kann zum Setzen der Maus an eine bestimmte Stelle mißbraucht werden. Wie? Das wissen Sie ja jetzt...

SH

```
 1:  /*****************************************************/
 2:  /*                                                   */
 3:  /*    Dies ist ein Demonstration der Gem-Routinen    */
 4:  /*             APPL_TRECORD und APPL_TPLAY           */
 5:  /* die dem Aufzeichnen und Abspielen von Ereignissen */
 6:  /* dienen                                            */
 7:  /*****************************************************/
 8:
 9:  char buffer[8*1000];
10:
11:  main()
12:  {
13:     int ap_ret;
14:
15:     appl_init();
16:
17:     printf("Bitte zeichnen Sie jetzt auf !
          (Taste drücken)\n");
18:     gemdos(7);
19:
20:  /* Aufzeichnen von 1000 Ereignissen */
21:
22:     ap_ret= appl_trecord(buffer,1000);
23:     printf("Alle Daten sind aufgezeichnet worden...\n");
24:
25:     printf("Jetzt wird wiedergegeben....
          (Taste drücken)\n");
26:     gemdos(7);
27:
28:     appl_tplay(buffer,1000,5);gemdos(7);
29:     appl_tplay(buffer,1000,10);gemdos(7);
30:     appl_tplay(buffer,1000,20);gemdos(7);
31:     appl_tplay(buffer,1000,30);gemdos(7);
32:
33:     appl_exit();
34:  }
```

```
 1:  /*****************************************************/
 2:  /*                                                   */
 3:  /*    Dies ist ein Demonstration der Gem-Routine     */
 4:  /*                 APPL_TPLAY                        */
 5:  /*    die dem Abspielen von Ereignissen dient.       */
 6:  /*              ( Zeitsteuerung )                    */
 7:  /*****************************************************/
 8:
 9:  int buffer[]={
10:  0,2,10,10,            /* Maus (10,10 ) */
11:  0,0,0,100,            /* 10 Sekunden   */
12:  0,2,20,10,            /* Maus (20,10 ) */
13:  0,0,0,200,            /* 20 Sekunden   */
14:  0,2,30,10,            /* Maus (30,10 ) */
15:  0,0,0,50,             /* 5 Sekunden    */
16:  0,2,40,10             /* Maus (40,10 ) */
17:  };
18:  main()
19:  {
20:     int ap_ret;
21:     int i;
22:
23:     appl_init();
24:
25:     printf("Bitte starten Sie jetzt !
          (Taste drücken)\n");
26:     gemdos(7);
27:
28:     appl_tplay(buffer,7,1);gemdos(7);
        /* 1:1-Geschwindigkeit */
29:
30:     appl_exit();
31:  }
```

GFA BASIC BESTSELLER







VERDEUTLICHT AN MEHR ALS 50 PROGRAMMBEISPIELEN DIE PROGRAMMIERUNG IN GFA-BASIC – DEM WEITVERBREITETEN BASIC AUF DEM ATARI ST

ANWENDUNGEN IN GFA-BASIC

INHALT

- Alle Programme sind ausführlich dokumentiert und übersichtlich programmiert, sodaß sie für jedermann verständlich sind
- Zahlreiche kurze Listings geben Tips, Tricks und Anregungen, die in eigene Programme übernommen werden können
- Die Programmthemen sind praxisnah ausgewählt, entstammen vielen Bereichen und sind für jedermann von Interesse
- Zur Einsparung umfangreicher Tipparbeit gibt es die Programmdiskette mit allen GFA-BASIC-Programmen dieses Buches

WICHTIGE MERKMALE

- **Tips und Tricks zur GFA-BASIC-Programmierung** (u. a. Verwendung von GEM-Funktionen, Eingaberoutinen, Spriteprogrammierung)
- **Utility- und Hilfsprogramme** (u. a. Kopierprogramme, Mauszeiger, Editor, Sprite- und Füllmustereditor, Disk-Echekker)
- **Grafik-Programmierung in GFA-BASIC** (u. a. 3D-Grafik, Turtlegrafik)
- **Anwendungsprogramme** (u. a. Dateiverwaltung, Präsentation, Manager, Vokabeltrainer, Widerstanddekodierung)
- **Mathematische Anwendungen** (u. a. Statistik, Ableitungen, Simps Integration)
- **Spiele** (Alamo, Space-Race, Hamurabi, Klicker)

und vieles andere mehr!

- **Programmdiskette zum Buch mit über 50 z. T. umfangreichen Programmen.** D-430 DM 39,–

Über 300 Seiten
Best.-Nr. B-430 **DM 49,–***

DAS GFA-BASIC BUCH

INHALT

- Der Umgang mit dem Editor
- Die Vorteile der strukturierten Programmierung ohne Zeilennummern und Pascal-ähnliche Struktur
- Kommunikation mit der Außenwelt (Peripherie)-Floppy, Tastatur-Prozessor, MIDI, der parallele u. serielle Bus, Joystick und Maus
- Mathematik und hohe Genauigkeit
- Schnelle Graphik schnell programmiert – Windows, Sprites, Alertbox, Pulldown Menüs
- Die Systembefehle – einfaches Einbinden von TOS-Befehlen (XBIOS, BIOS, GEMDOS) zur vollständigen Nutzung des Atari-Betriebssystems
- Der Sound – Beschreibung des Soundchips und dessen Programmierung, sogar mit Interrupt-Steuerung

WICHTIGE MERKMALE

- Die weit über 200 Befehle des neuen GfA-Basic-Interpreters werden mit ihrer Syntax und Bedeutung anhand von vielen Beispielen ausführlich erklärt.
- Das GfA-Basic-Buch ist in Sachgebiete unterteilt, um eine sinnvolle Anwendung der Befehle im Zusammenhang zu ermöglichen. Dabei werden die vielfältigen Fähigkeiten der ATARI ST-Rechner (Graphik, Geschwindigkeit, Window-Technik und Sound) genutzt.
- Das GfA-Basic-Buch ist leicht verständlich geschrieben, so daß auch der Anfänger ohne Probleme seine eigenen Programme erstellen kann.
- Aber auch für den fortgeschrittenen Anwender und für den, der es einmal werden will, bietet das GfA-Basic-Buch die Grundlage zur richtigen Handhabung des Betriebssystems.
- Im Anhang ist eine komplette alphabetische Befehlsübersicht (Nachschlagewerk) enthalten.
- **Programmdiskette zum Buch: DM 39,–**

Über 500 Seiten **DM 49,–***

DIE GROSSE GFA-BASIC PROGRAMM-SAMMLUNG

INHALT

Folgende Schwerpunkte sind gesetzt: Allgemeine Eigenschaften des GFA-Basic · Zahlen- und Variablenbehandlung · Dateiverwaltung und Ordner in GFA-Basic · Grundlegende Befehle · Schleifentechnik · Felder (Tabellenverarbeitung u. Matrizen) · Unterprogramme · Menü-Steuerung · Window-Technik · Zufallszahlen · Sequentielle Dateien · RANDOM-Dateien · Physikalischer Diskettenzugriff · Textverarbeitung · Sortierprogramme · Sortiersystem: SORTMIX · Fakturiersystem: FAKTURA · Grafik auf dem Bildschirm u. dem Drukker · Grafik-Programm: GRAFIKER · Geometrie, Statistik u. Funktionsplott · Bildschirmspiel: FLUSS · Kontoführungsprogramm: KONTO 520 · und vieles mehr

WICHTIGE MERKMALE

Anhand von über 130 Programmbeispielen und Erläuterungen lernen und trainieren Sie das Programmieren in GFA-BASIC. Von einfachen, aber grundlegenden Beispielen bis zur ausgereiften Anwendung findet der ST-Besitzer Beispiel-Programme, die den perfekten Einstieg in die Programmiersprache GFA-BASIC leicht und interessant machen.
Alle Beispiele werden so dargestellt, daß das Verständnis für den Programmaufbau gefördert wird, aber auch die Details der GFA-BASIC-Version deutlich werden. Die Programme sind gut dokumentiert, wobei die hervorragenden Möglichkeiten des GFA-BASIC zur strukturierten Programmierung genutzt werden.

- **Programmdiskette zur Progr.-Sa.: DM 39,– mit 131 Programmen in GFA-BASIC**

320 Seiten **DM 49,–***

* Alle Preise sind unverbindlich empfohlene Verkaufspreise

Heim Verlag
Heidelberger Landstr. 194
6100 Darmstadt-Eberstadt
Telefon 06151-56057

BESTELL-COUPON

an Heim-Verlag
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt

Ich bestelle ____________________
zzgl. DM 5,– Versandkosten (unabhängig von der bestellten Stückzahl)
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei

Name, Vorname ____________________

Straße, Hausnummer ____________________

PLZ, Ort ____________________
Benutzen Sie auch die im ST COMPUTER vorhandene Bestellkarte.

```
 1:  /**************************************************/
 2:  /*                                                */
 3:  /*    Dies ist ein Demonstration der Gem-Routine  */
 4:  /*                  APPL_TPLAY                    */
 5:  /*    die dem Abspielen von Ereignissen dient.    */
 6:  /*               ( Skalierung )                   */
 7:  /**************************************************/
 8:
 9:  int buffer[8*100];
10:
11:  main()
12:  {
13:     int ap_ret;
14:     int i;
15:
16:     appl_init();
17:
18:     for (i=0; i<100*8; i+=8)
19:     {
20:                  /* Maussteuerung */
21:        buffer[i]=0;       /* Maus- */
22:        buffer[i+1]=2;     /* Ereignis */
23:        buffer[i+2]=i/4;   /* X-Inkrement */
24:        buffer[i+3]=50;    /* Y-Koordinate konstant */
25:
26:        buffer[i+4]=0;     /* Zeit- */
27:        buffer[i+5]=0;     /* Ereignis */
28:        buffer[i+6]=0;
29:        buffer[i+7]=10;    /* Zeit = 10  */
30:
31:     }
32:
33:     printf("Bitte starten Sie jetzt !
          (Taste drücken)\n");
34:     gemdos(7);
35:
36:     appl_tplay(buffer,200,1);gemdos(7);        /* 1:1 */
37:     appl_tplay(buffer,200,2);gemdos(7);        /* 1:2 */
38:     appl_tplay(buffer,200,3);gemdos(7);        /* 1:3 */
39:     appl_tplay(buffer,200,4);gemdos(7);        /* 1:4 */
40:     appl_tplay(buffer,200,5);gemdos(7);        /* 1:5 */
41:     appl_tplay(buffer,200,10);gemdos(7);       /* 1:10  */
42:
43:     appl_exit();
44:  }
```

```
 1:  /**************************************************/
 2:  /*                                                */
 3:  /*     Dies ist ein Demonstration der Gem-Routine */
 4:  /*                 APPL_TRECORD                   */
 5:  /*     die dem Aufzeichnen von Ereignissen dient  */
 6:  /*                                                */
 7:  /**************************************************/
 8:
 9:  char buffer[8*10];
10:
11:  main()
12:  {
13:     int ap_ret;
14:     char i;
15:
16:     appl_init();
17:
18:     printf("Bitte zeichnen Sie jetzt auf !
          (Taste drücken)\n");
19:     gemdos(7);
20:
21:  /* Aufzeichnen von 10 Ereignissen */
22:
23:     ap_ret= appl_trecord(buffer,10);
24:     printf("Alle Daten sind aufgezeichnet worden...\n");
25:
26:  /* Ausgeben des Buffers auf den Bildschirm */
27:
28:     for (i=0; i<80; i+=2)
29:     {
30:
31:        if (!(i%16))
32:           printf("\n");
33:
34:        printf("%4x ",*(int*)(buffer+i));
35:     }
36:     gemdos(7);
37:
38:     appl_exit();
39:  }
```

Harte Sachen ! - Easytizer zu gewinnen !

Wir möchten dem häufig geäußerten Wunsch unserer Leser, mehr Hardware zu bringen, in der Zukunft größere Aufmerksamkeit widmen. Das erste Projekt, der Easytizer, ist ja schon veröffentlicht, und es hat sich eine große Resonanz gezeigt.

Um nun lesernahe Projekte entwickeln zu können, möchten wir alle unsere Leser bitten, uns Ihre Wünsche, Probleme oder Ideen, die sich als Projekt veröffentlichen lassen, mitzuteilen. Dazu kann man die im Heft befindliche Postkarte an die Redaktion verwenden und mit dem Stichwort 'Hardware' versehen. Natürlich sind auch fertige Schaltungsideen jederzeit willkommen.

Damit sich das Schreiben auch lohnt, verlosen wir unter den Einsendungen, gleichgültig ob es sich um ein Angebot, eine Idee oder auch nur ein Wunsch handelt, ab Dezember 1987 alle zwei Monate einen fertig aufgebauten Easytizer.

'MERLIN' Computer GmbH
'Hardware'
Industriestr. 26
D-6236 Eschborn



ABO

ABO

Absender
(Bitte deutlich schreiben)

Vorname/Name

Straße/Nr.

PLZ/Ort

Bitte
mit
60 Pf.
frankieren

Postkarte

Heim-Verlag
Heidelberger Landstr. 194

6100 Darmstadt-Eberstadt
Telefon 0 61 51 / 5 60 57



Einzelheft- u. Monatsdisketten Bestellung



Einzelheft- u.
Disketten Service

Absender
(Bitte deutlich schreiben)

Vorname/Name

Straße/Nr.

PLZ/Ort

Bitte
mit
60 Pf.
frankieren

Postkarte

Heim-Verlag
Heidelberger Landstr. 194

6100 Darmstadt-Eberstadt
Telefon 0 61 51 / 5 60 57



Kleinanzeigen



Kleinanzeigen

Absender
(Bitte deutlich schreiben)

Vorname/Name

Straße/Nr.

PLZ/Ort

Bitte
mit
60 Pf.
frankieren

Postkarte

Heim-Verlag
Heidelberger Landstr. 194

6100 Darmstadt-Eberstadt
Telefon 0 61 51 / 5 60 57



Abonnement

Ja, bitte senden Sie mir die ATARI-Computer Fachzeitschrift ab ______________________
für mindestens 1 Jahr (11 Hefte) zum ermäßigten Preis von jährlich DM 70,– frei Haus.
(Ausland: Nur gegen Scheck-Voreinsendung DM 90,– Normalpost. DM 120,– Luftpost)
Der Bezugszeitraum verlängert sich nur dann um ein Jahr, wenn nicht 6 Wochen vor Ablauf des Abonnements gekündigt wird.

Name ______________________

Vorname ______________________

Straße/Nr. ______________________

PLZ ______ Ort ______________________

Gewünschte Zahlungsweise bitte ankreuzen

☐ Bequem und bargeldlos durch Bankeinzug

Konto-Nr. ____________ BLZ ____________

Institut ____________ Ort ____________

☐ Ein Verrechnungsscheck über DM ______ liegt bei.

☐ Vorauskasse per Post-Einzahlung (Zahlkarte)

Garantie:
Diese Bestellung kann ich schriftlich innerhalb einer Woche (rechtzeitige Absendung genügt) widerrufen. Dies bestätige ich durch meine 2. Unterschrift.

Datum ____________ Unterschrift ____________

Datum ____________ Unterschrift ____________

COMPUTER

ABO



Einzelheft- u. Monatsdisketten **Bestellung**

ST-Computer können Sie direkt beim HEIM-VERLAG zum Einzelheft-Preis von DM 6,— (1986 + 1987) DM 7,— (1988) nachbestellen. Bearbeitung nur gegen beigefügten Scheck über den entsprechenden Betrag (keine Überweisung).

Jan.	Febr.	März	April	Mai	Juni	Juli/Aug.	Sept.	Okt.	Nov.	Dez.	
											1986 = DM
											1987 = DM
											1988 = DM

+ Gebühr für Porto u. Verp. = DM

(1 Heft DM 2,–, ab 2 Hefte DM 5,–)

☐ Scheck in Höhe zus. DM liegt bei

Disketten Service

Alle Programme, die in ST-Computer veröffentlicht wurden, sind auf Disketten erhältlich. Die Disketten enthalten die Programme von jeweils 2 ST-Computer-Ausgaben. Bestellen Sie durch ankreuzen die gewünschten Disketten

Preis je Diskette 28,– DM	Jan./Febr.	März/Apr.	Mai/Juni	Juli/Aug.	Sept./Okt.	Nov./Dez.	Jan./Febr.	März/Apr.	Mai/Juni
	87	87	87	87	87	87	88	88	88

Lieferung: gegen beigefügten Scheck zuzügl. 5,— DM Versandkosten, unabhängig der bestellten Menge.







Kleinanzeigen-Auftrag

Bitte veröffentlichen Sie für mich folgende Kleinanzeige in der angekreuzten Rubrik

Biete an ☐ Hardware ☐ Software **Ich suche** ☐ Hardware ☐ Software ☐ **Tausch** ☐ **Kontakte** ☐ Verschiedenes ☐

30 Buchstaben je Standardzeile – incl. Satzzeichen und Wortzwischenräume.
Groß- und Kleinbuchstaben verwenden, fettgedruckte Wörter unterstreichen.

____
____
____
____
____
____

Bearbeitung **nur gegen Vorausscheck** über den entsprechenden Betrag (keine Überweisung)

☐ privat = DM 7,– je Zeile incl. MwSt.
☐ gewerblich = DM 15,– je Zeile + MwSt.
☐ Chiffregebühr = DM 10,–

☐ Scheck über DM ____________ ist beigefügt

Bei Angeboten: Ich bestätige, daß ich alle Rechte an den angebotenen Sachen besitze.

Datum ____________ Unterschrift ____________

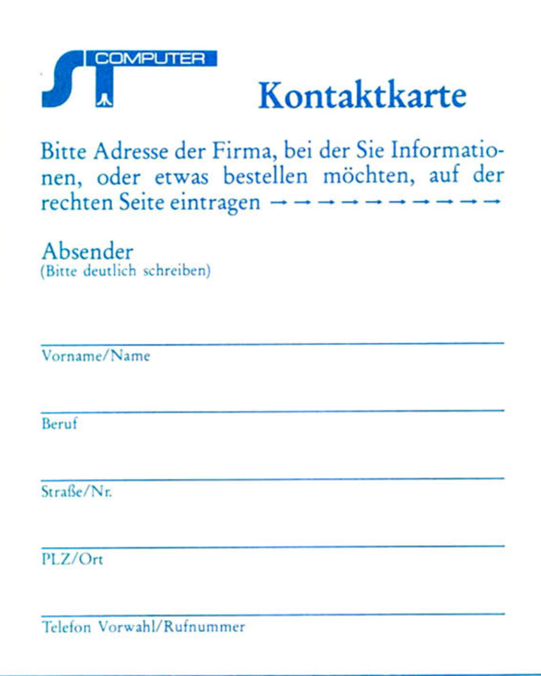
Absenderangaben auf der Rückseite nicht vergessen



Kleinanzeigen



Kontaktkarte



ST COMPUTER

Kontaktkarte

Bitte Adresse der Firma, bei der Sie Informationen, oder etwas bestellen möchten, auf der rechten Seite eintragen → → → → → → → → → →

Absender
(Bitte deutlich schreiben)

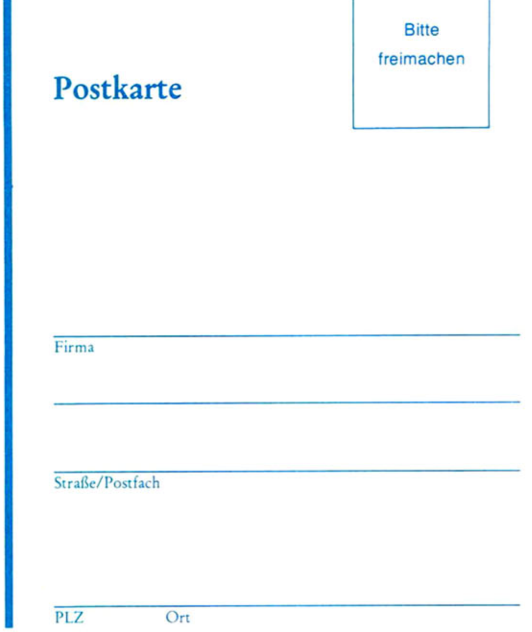
Vorname/Name

Beruf

Straße/Nr.

PLZ/Ort

Telefon Vorwahl/Rufnummer



Postkarte

Bitte freimachen

Firma

Straße/Postfach

PLZ Ort



Kurzmitteilung



ST COMPUTER

Kurzmitteilung

Absender
(Bitte deutlich schreiben)

Vorname/Name

Straße/Nr.

PLZ/Ort

Telefon

Postkarte

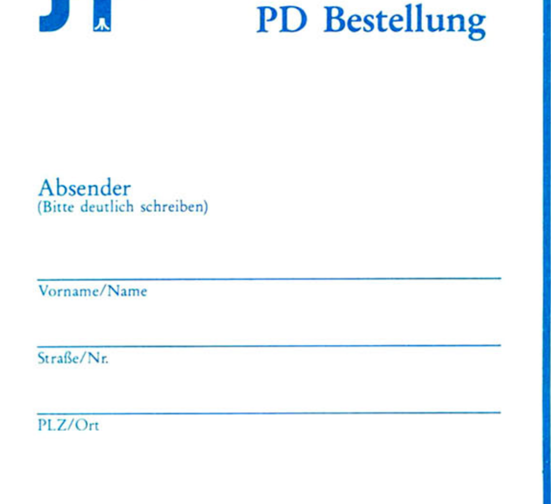
Bitte freimachen

Merlin Computer GmbH
ST-Computer Redaktion
Industriestraße 26

6236 Eschborn



PD Bestellung



ST COMPUTER

PD Bestellung

Absender
(Bitte deutlich schreiben)

Vorname/Name

Straße/Nr.

PLZ/Ort

Postkarte

Bitte freimachen

Merlin Computer GmbH
ST-Computer Redaktion
Industriestraße 26

6236 Eschborn

ST COMPUTER

☐ Ich bitte um weitere Informationen

☐ Ich gebe folgende Bestellung auf

in Bezug auf Ihre Anzeige in ST-Computer Heft ______ Seite ___

Menge	Produkt/Bestellnummer	DM	gesamt DM

Datum, Unterschrift (für Jugendliche unter 18 Jahren der Erziehungsberechtigte)

Abgesandt am:

ST COMPUTER

Firma:

Bemerkungen:

ST COMPUTER

Meine Meinung

Zu dem Artikel ____________ in Heft ____, Seite ____ hätte ich folgendes zu bemerken:

☐ Ich möchte Ihnen folgendes Programm zur Veröffentlichung anbieten: (Kurzbeschreibung, Sprache, Länge in Druckerseiten, GEM/TOS)

☐ Ich kann über folgendes Thema berichten: (Tips & Tricks am ST, Hardware, Software, etc.)

☐ Ich möchte gerne Autor in der ST-Computer werden. Meine Fachgebiete: (z.B. LISP, Pearl, Modula-2, Assembler ...)

☐ Ich möchte, daß folgendes Public-Domain Programm in Ihre Sammlung aufgenommen wird.

☐ Sonstiges

Bei weiteren Angaben oder Fragen wenden Sie sich bitte schriftlich oder telefonisch an die Redaktion. Tel. 0 61 96/48 18 11

ST COMPUTER

Kurzmitteilung

ST COMPUTER

PUBLIC DOMAIN SERVICE

Ich bestelle folgende PD-Disketten:
(Siehe PD Service in dieser Ausgabe)

Zahlung erfolgt:

☐ per Scheck

☐ per Nachnahme

Je Diskette fügen Sie bitte einen Betrag von DM 10,– bei, für Porto und Verpackung je Sendung DM 5,– (Ausland DM 10,–)

Datum

Unterschrift

ST COMPUTER

PD Bestellung

Datum

Einkaufsführer

Hier finden Sie Ihren Atari Fachhändler

Anzeigenschluß für Heft 6/88 ist der 23.04.88

1000 Berlin



Computare

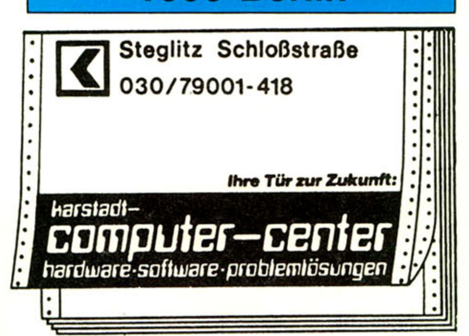
Keithstr. 18-20 • 1000 Berlin 30
030/21 390 21
186 346 com d







1000 Berlin





Computershop Edith Behrendt

Fürbringerstraße 26 · 1000 Berlin 61
Telefon 0 30/691 76 66





2000 Hamburg

Bit Computer Shop
Osterstraße 173 · 2000 Hamburg 20
Telefon: 040/49 44 00

Createam
Computer Hard & Software
Bramfelder Chaussee 300 · 2000 Hamburg 71
Telefon Sa. Nr. 040/641 50 91

Hardware
Software
Beratung
Service



ATARI Systemfachhändler
Münsterstraße 9 · 2000 Hamburg 54
Telefon 040/56 60 1-1

Gerhard u. Bernd Waller GbR
Computer & Zubehör-Shop
Kieler Straße 623
2000 Hamburg 54
040/570 60 07 + 570 52 75

NEU: Software Shop
RADIX Bürotechnik
Heinrich Barth Str. 13
2000 Hamburg 13
Telefon: 0 40-44 16 95



2000 Norderstedt



2120 Lüneburg

Sienknecht
Bürokommunikation
Beratung - Verkauf - Werkstatt
Heiligengeiststr. 20, 2120 Lüneburg
Tel. 04131/46122, Btx 402422
Mo.-Fr. 9°°-18°° und Sa. 9°°-13°°

4000 Düsseldorf

HOCO
EDV ANLAGEN GMBH
Flügelstr. 47
4000 Düsseldorf
Tel. 02 11 - 77 62 70

4050 Mönchengladbach



Hindenburgstr. 249
4050 Mönchengladbach
Tel. 0 21 61 - 187 64

4150 Krefeld



- Service-Center
- Hardware
- Software
- Erweiterungen

COP
Computer Service GmbH
Lewerentz Str. 111
4150 Krefeld
Tel. 0 21 51 / 77 30 42

4250 Bottrop

Megateam-Computer-Systeme

Kirchhellenerstraße 262
4250 Bottrop

4300 Essen

ATARI Systemfachhändler



KARSTADT Aktiengesellschaft
Limbecker Platz 4300 Essen 1
Tel.: (02 01) 17 63 99

4320 Hattingen



4330 Mülheim

BNS Computer und Bürotechnik
Vertriebsgesellschaft mbH
Dickswall 79 4330 Mülheim Telefon 02 08/3 40 34

Computer Hard- und Software auch im Leasing
Computerkurse für Anfänger und Fortgeschrittene

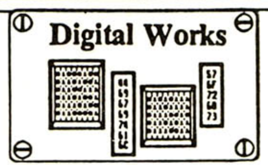


VICTOR SEL-Fernkopierer
NEC ATARI OKI

4400 Münster



4410 Warendorf



Computer-Fachhandel – Hardware & Software

Jörg Kellert – Helmut Müller GbR
Brünebrede 17 · 4410 Warendorf
Tel. 0 25 81 / 6 11 26

4422 Ahaus

ATARI · Epson · Fujitsu
Molecular · NCR · Tandon · Schneider · Star

OCB

OCB-Computershop	OCB-Hard- und Software
Wallstraße 3	Wessumerstraße 49
4422 Ahaus	4422 Ahaus
Tel. 0 25 61/50 21	Tel. 0 25 61/50 21

4430 Steinfurt

ATARI SCHNEIDER STAR NEC SEIKOSHA PANASONIC EPSON

Computer
Büromaschinen
Service

Telefon 02551/2555

Tecklenburger Str. 27 · 4430 Steinfurt

4500 Osnabrück

Heinicke-Electronic

Kommenderiestr. 120·4500 Osnabrück
Telefon 05 41 - 8 27 99

Wir liefern Micro-Computer seit 1978

4600 Dortmund

Bürostudio BOLZ

Brauhausstraße 4 · 4600 Dortmund
Telefon 02 31 - 52 77 13 - 16



4600 Dortmund



Atari, Genie, Schneider, Tandy, Brother, Star, Memorex, BASF, Verbatim

cc Computer Studio GmbH
Software-Hardware-Beratung
Service-Eilversand

Ihre Ansprechpartner:
v. Schablinski
Jan P. Schneider

Elisabethstraße 5
4600 Dortmund 1
T. 0231/528184 · Tx 822631 cccsd

Elektronik
Computer
Fachliteratur

ATARI-System-Fachhändler

4600 Dortmund 1, Güntherstraße 75, Tel. (02 31) 57 22 84



4620 Castrop-Rauxel

R. Schuster Electronic
OBERE MÜNSTERSTR. 33-35 · (0 23 06) 3770 · 4620 CASTROP-RAUXEL



System-Fachhändler

4650 Gelsenkirchen-Horst



Hard- und Software, Literatur
Bauteile, Service, Versand
Groß- und Einzelhandel
Poststr. 15 · 4650 Gelsenkirchen-Horst
Tel. 0209/52572

4700 Hamm

computer center



4800 Bielefeld

hardware
software
organisation
service



CSF COMPUTER & SOFTWARE GMBH
Heeper Straße 106 – 108
4800 Bielefeld 1
Tel. (05 21) 6 16 63

Carl-Severing-Str. 190
4800 Bielefeld 14
Telefon: 05 21/45 99-150
Telex : 9 37 340 krab d
Telefax: 05 21/45 99-123

MICROTEC
Software
Hardware
Beratung
Service

5000 Köln

BÜRO MASCHINEN
braun
AM RUDOLFPLATZ GmbH
5000 KÖLN 1
RICHARD-WAGNER-STR. 39
RUF: 02 21/ 21 91 71

5010 Bergheim



EDV-Beratung · Organisation
Programmierung · Home/Personal-Computer
Software · Zubehör · Fachliteratur
Zeppelinstr. 7 · 5010 Bergheim
Telefon 0 22 71 - 6 20 96

5060 Bergisch-Gladbach

Computer Center
Buchholzstraße 1
5060 Bergisch-Gladbach
Telefon 0 22 02 - 3 50 53

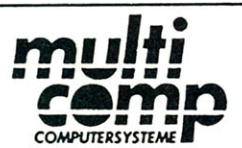
5090 Leverkusen

Rolf Rocke
Computer-Fachgeschäft
Auestraße 1
5090 Leverkusen 3
Telefon 0 21 71 / 26 24

5200 Siebgurg

Computer Center
Luisenstraße 26
5200 Siegburg
Telefon 0 22 41/6 68 54

5220 Waldbröl



Waldstraße 1 · 5220 Waldbröl
☎ (0 22 91) 44 08/33 86

5300 Bonn



Gesellschaft für Computer- und Communicationstechnologie mbH
Hardware · Software · EDV-Zubehör
Telefon 02 28/22 24 08
COCO GmbH · Schumannstraße 2 · 5300 Bonn 1

5457 Straßenhaus

DR. AUMANN GMBH
Computer-Systeme
Schulstr. 12
5457 Straßenhaus
Telefon 0 26 34 - 40 81/2

5500 Trier



bürocenter
LEHR
Güterstr. 82 - 5500 Trier
☎ 06 51 - 2 50 44
Fordern Sie unsere Zubehör-Liste an.

5540 Prüm

ATC COMPUTER
J. M. ZABELL
Ritzstraße 13 · Pf. 10 51
5540 PRÜM
– Tel.: 0 65 51 - 30 39 –

5600 Wuppertal

Jung am Wall
Wall 31—33
5600 Wuppertal 1
Telefon 02 02/45 03 30

MEGABYTE
Computer Vertriebs GmbH
Friedrich-Engels-Allee 162
5600 Wuppertal 2 (Barmen)
Telefon (02 02) 8 19 17

5630 Remscheid

C O M SOFT
Scheiderstr. 12 · 5630 Remscheid
Telefon (0 21 91) 2 10 33 - 34

5650 Solingen

MegaTeam
Hardware – Software
Zubehör – Service
Hauptstraße 1 – 3
5650 Solingen 1
Tel. 02 12/4 58 88

5800 Hagen



Vertragshändler **Axel Böckem**
Computer + Textsysteme
Eilper Str. 60 (Eilpezentrum) · 5800 Hagen
Tel. 0 23 31 / 7 34 90

5900 Siegen



Siegen · Weidenauer Str. 72 · ☎ 02 71/7 34 95

6000 Frankfurt

Müller & Nemecek
Kaiserstraße 44
6000 Frankfurt/M.
Tel. 0 69 - 23 25 44

WAIZENEGGER
Büroeinrichtungen
Kaiserstraße 41
6000 Frankfurt/M.
☎ 0 69 / 27 30 06 - 0



Büro-Computer + Organisations GmbH
Oederweg 7-9
6000 Frankfurt am Main 1
☎ (0 69) 55 04 56 - 57
Commodore OKI ATARI TOSHIBA

6100 Darmstadt

Heim
Büro- und Computermarkt
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt
✆ 0 61 51 / 5 60 57

ATARI Systemfachhändler



KARSTADT Aktiengesellschaft
Elisabethenstr. 15 · 6100 Darmstadt
Luisencenter · Tel. 0 61 51 - 10 94 20

6240 Königstein

KFC
COMPUTERSYSTEME
Wiesenstraße 18
6240 Königstein
Tel. 0 61 74 - 30 33
Mail-Box 0 61 74 - 53 55

6300 Gießen



Schneider ATARI Commodore

BAUMS
BÜRO · ORGANISATION
Bahnhofstr. 26 · 6300 Gießen
Telefon: 06 41 / 7 10 96



Thomas Heß
Asterweg 10 · 6300 Gießen
Telefon 06 41 / 3 91 53

6330 Wetzlar



Fachmarkt für Computer u. Unterhaltungs-electronic in Wetzlar,
Einkaufszentrum Bahnhofstraße, Tel. (0 64 41) 4 85 66

6400 Fulda

Schneider ATARI Commodore

WEINRICH
BÜRO · ORGANISATION
Ronsbachstraße 32 · 6400 Fulda
Telefon: 06 61 / 4 92 - 0

6457 Maintal

Landolt-Computer
Beratung · Service · Verkauf · Leasing

Wingertstr. 112
6457 Maintal/Dörnigheim
Telefon 0 61 81 - 4 52 93

6500 Mainz

:ELPHOTEC
Computer Systeme
Ihr Atari Systemhändler mit eigenem Service-Center
Walpodenstraße 10
6500 Mainz
Telefon 0 61 31 - 23 19 47

6520 Worms

ORION
Computersysteme GmbH
Friedrichstraße 22
6520 WORMS
Tel. 0 62 41 / 67 57 - 67 58

6700 Ludwigshafen

MKV Computermarkt
Bismarck-Zentrum
6700 Ludwigshafen
Telefon 06 21 - 52 55 96

6720 Speyer

MKV Computermarkt
Gilgenstraße 4
6720 Speyer
Telefon 0 62 32 - 7 72 16

6750 Kaiserslautern

C.O.S
Computer Organisation GmbH
Friedrichstr. 4 · 6750 Kaiserslautern
Telefon (06 31) 9 20 51

6800 Mannheim



Computersysteme + Textsysteme
6800 Mannheim 24
Casterfeldstraße 74-76
☎ (06 21) 85 00 40 · Teletex 6 211 912

Computer-Center
am Hauptbahnhof GmbH
L 14, 16-17
6800 Mannheim 1
Tel. (06 21) 2 09 83 / 84

6900 Heidelberg

JACOM COMPUTERWELT
Hardware · Software
Schulung · Service
Mönchhofstraße 3 · 6900 Heidelberg
Telefon 0 62 21 / 41 05 14 - 550

HEIDELBERGER COMPUTER CENTER
Bahnhofstraße 1
6900 Heidelberg
Telefon 06221/27132

7022 L-Echterdingen

Autorisierter ATARI-System-Fachhändler
ATARI ST



Matrai Computer GmbH
Bernhauser Str. 8
7022 L.-Echterdingen
☎ (07 11) 79 70 49

7030 Böblingen

Verkauf – Service – Software

Norbert Hlawinka
Sindelfinger Allee 1
7030 Böblingen
Tel. 0 70 31 / 22 60 15



COMPUTER CENTER

7100 Heilbronn

Unser Wissen ist Ihr Vorteil

Walliser & Co.
Mönchseestraße 99
7100 Heilbronn
Telefon 07131/60048



Computer-Welt
Am Wollhaus 6
7100 Heilbronn
Tel. 0 71 31 - 6 84 01 - 02

7101 Flein

der **COMPUTERLADEN** von



In der Falterstraße
7101 Flein
Beratung, Service, Verkauf,
Software-Entwicklung
direkt beim Peripherie-Hersteller

7150 Backnang



7400 Tübingen

Werner Brock
COMPUTERSSTUDIO
Poststraße 2 – 4 · D-7400 Tübingen
Tel. 0 70 71 - 3 43 48 · Fax 0 70 21 - 3 47 92

Autorisierter Systemfachhändler für:
ATARI, Schneider, Commodore, Panasonic, Kaypro, Sharp, NEC, OKI, STAR,...

7410 Reutlingen

Werner Brock
COMPUTERSHOP
Federnseestr. 17 · 7410 Reutlingen
Tel. 07121-34287
Tx 172414024 RMI D · box:rmi:taisoft · Fax 07121-339779
Autorisierter Systemfachhändler für:
ATARI, Schneider, Commodore, Panasonic, Kaypro, Sharp, NEC, OKI, STAR,...

7450 Hechingen



Gesellschaft für Datenverarbeitung mbh
Computer · Drucker
Zubehör · Fachliteratur
Schloßplatz 3 · 7450 Hechingen
Telefon 07471/14507

7475 Meßstetten

Ihr ATARI-Systemhändler im Zollern-Alb-Kreis
HEIM + PC-COMPUTERMARKT



HARDWARE · SOFTWARE · LITERATUR
ATARI COMMODORE CUMANA DATA-BECKER MULTITECH RITEMAN SCHNEIDER THOMSON
7475 Meßstetten 1 · Hauptstraße 10 · 07431/61280

7480 Sigmaringen

Wir stellen uns für Sie auf den Kopf!
Wir führen:
Hardware - Software - Zubehör
Zeitschriften - Bücher
Wir sind autorisierte Fachhändler von:
ATARI Schneider
Tandon
SOFT&EASY
COMPUTER GMBH
7480 Sigmaringen Am Rappgasse Tel 07571/12483
7968 Saulgau Pfarrstraße 13 Tel 07581/2598

7500 Karlsruhe

papierhaus erhardt
Am Ludwigsplatz · 7500 Karlsruhe
Tel. 0721-1608-16

MKV GMBH
Kriegsstraße 77
7500 Karlsruhe
Telefon 0721-84613

7530 Pforzheim

DM Computer GmbH
Hard- & Software
Durlacherstr. 39
Tel. 07231-13939
7530 Pforzheim
Telex 783248

7600 Offenburg

FRANK LEONHARDT ELECTRONIC
Ihr Fachgeschäft für Microcomputer · Hifi · Funk
In der Jeuch 3
7600 Offenburg
Telefon 0781/57974

7640 Kehl/Rhein



Badstrasse 12
Tel. 07807/822
Telex: 752913
7607 NEURIED 2
Filiale:
Hauptstrasse 44
Tel. 07851/1822
7640 KEHL/RHEIN
ELEKTRO-MÜNTZER GmbH

7700 Singen

U. MEIER
Computersysteme
7700 Singen-Htwl.
Am Posthalterswäldle 8
Telefon 07731-44211

7730 VS-Schwenningen

BUS BRAUCH & SAUTER
COMPUTER TECHNIK
Villinger Straße 85
7730 VS-Schwenningen
Telefon 07720/38071-72

7750 Konstanz

ATARI ★ PC's ★ SCHNEIDER
computer-fachgeschäft
rösler
Rheingutstr. 1 · 07531-21832

7800 Freiburg

CDS
EDV-Service GmbH
Windausstraße 2
7800 Freiburg
Tel. 0761-81047

7800 Freiburg



KARTÄUSERSTRASSE 59
D-7800 FREIBURG/BRST.
TELEFON 0761-382038

7890 Waldshut-Tiengen

hettler-data
service gmbh
Lenzburger Straße 4
7890 Waldshut-Tiengen
Telefon 07751/3094

7900 Ulm

HARD AND SOFT COMPUTER GMBH
Ulms großes Fachgeschäft
für BTX, Heim- u. Personalcomputer
Herrenkellergasse 16 · 7900 Ulm/Donau
Telefon 0731/62699

EDV-Systeme
Software-erstellung
Schulung

Systemhaus:
Frauenstr. 28
7900 Ulm/Donau
Tel. 0731/28076
Telex 712973 csulm-d



7918 Illertissen

biotech gmbh
technische Informationsysteme
Computerladen
Marktplatz 13
7918 Illertissen
07303/5045

7950 Biberach

HARD AND SOFT COMPUTER GMBH
Biberachs großes Fachgeschäft
für BTX, Heim- u. Personalcomputer
Schulstraße 6 · Bei der VHF
7950 Biberach · Tel. 07351/12221

7980 Ravensburg

GRAHLE
Expert Grahle Computer
Eisenbahnstr. 33
7980 Ravensburg
Tel.: 0751/15955
Vertragshändler für ATARI, Schneider und Star

8000 München

Ludwig
COMPUTER + BÜROTECHNIK
COMPUTER · SOFTWARE · PERIPHERIE
BERATUNG · TECHN. KUNDENDIENST
INGOLSTÄDTER STR. 62L
EURO-INDUSTRIE-PARK · 8000 MÜNCHEN 45
TELEFON 089/3113066 · TELETEX 898341

schulz computer
Schillerstraße 22
8000 München 2
Telefon (089) 597339
Beratung · Verkauf · Kundendienst

8032 Gräfeling



COMPUTER SYSTEME SCHULUNG
Am Haag 5
8032 Gräfelfing
Tel. 089-8545464,851043

8100 Garmisch-Part.

Uwe Langheinrich
Elektronik Center
Hindenburgerstr. 45
8100 Garmisch-Partenkirchen
Tel. 08821-71555
Bitte Gratisliste anfordern

8120 Weilheim

Klement
COMPUTER FACHHÄNDLER
Elektro-, Radio- und Fernseh-Center
Beleuchtungskörper · Schallplatten
Meisterbetrieb
8120 Weilheim · Admiral-Hipper-Straße 1
Geschäft ✆ 4500 · Kundendienst ✆ 4400
Computer-Studio ✆ 62167
Interfunk-Fachgeschäft

8150 Holzkirchen

ATARI
Besuchen Sie uns!
Fordern Sie unseren Software-Katalog (520ST) an!


MÜNZENLOHER GMBH
Tölzer Straße 5
D-8150 Holzkirchen
Telefon: (08024) 1814

8170 Bad Tölz

Uwe Langheinrich
Elektronik Center
Wachterstr. 3
8170 Bad Tölz
Tel. 08041-41565
Bitte Gratisliste anfordern

8330 Eggenfelden

Hot Space
Computer-Centrum
R. Lanfermann
Schellenbruckstraße 6
8330 Eggenfelden
Telefon 08721/6573
Altöttinger Straße 2
8265 Neuötting
Telefon 08671/71610
Innstraße 4
8341 Simbach
Telefon 08571/4410

8400 Regensburg

Zimmermann elektroland
8400 Regensburg
Dr.-Gessler-Str. 8
☎ 0941/95085

8390 Passau
Kohlbruck 2a
☎ 0851/52007

8423 Abensberg

COMPUTERVERSAND WITTICH
Tulpenstr. 16 · 8423 Abensberg
☎ 09443/453



8500 Nürnberg





8520 Erlangen



8520 Erlangen

Computerservice
Decker
Meisenweg 29 - 8520 Erlangen
Telefon 09131/42076

Zimmermann elektroland
8520 Erlangen
Nürnberger Straße 88
Tel. (09131) 34568

8500 Nürnberg
Hauptmarkt 17
Tel. (0911) 20798

8600 Bamberg



Bamberg · Tel. 0951/27808-09

8700 Würzburg

SCHOLL
BÜROTEAM
Hardware · Software
Service · Schulung
computer center
am Dominikanerplatz
Ruf (0931) 50488

8720 Schweinfurt

Uhlenhuth GmbH
Computer + Unterhaltungselektronik
Albrecht-Dürer-Platz 2
8720 Schweinfurt
Telefon 09721/652154

8900 Augsburg



Unser Plus: Beratung u. Service
Schwalbenstr. 1 · 8900 Augsburg-Pfersee
Telefon (0821) 528533 oder 528087
Computer Vertriebs- und Software GmbH

8900 Augsburg

ATARI Systemfachhändler



KARSTADT Aktiengesellschaft
Bürgermeister-Fischer-Str. 6-10
8900 Augsburg · Tel. (08 21) 31 53-416

8940 Memmingen

EDV-Organisation
Hard- + Software
Manfred Schweizer KG
Ulmer Str. 2, Tel. 08331/12220
8940 Memmingen

Österreich

A-1020 Wien

W + H.
Computerhandel Ges m.b.H.

Förstergasse 6/3/2 · 1020 Wien
Tel. 02 22-35 09 68

A-1040 Wien

Ihr ST-Fachhändler in Wien
Computer-Studio
Wehsner Gesellschaft m.b.H.

1040 Wien - Paniglgasse 18-20
Telefon 02 22-65 78 08, 65 88 93

A-8010 Graz



Schweiz

CH-1205 Geneve

PIMENT ROUGE
INFORMATIQUE S.A.
8, RUE DES MARAICHERS
1205 GENEVE TEL. 022/28 56 24

CH-1700 Fribourg

FRIDAT SA INFORMATIQUE
ehem.
Softy Hard's Computershop
Rte des Grives 4
1700 Granges-Paccot / Fribourg
0041 (0)37 26 66 28

CH-2503 Biel

UE
URWA Electronic

Ihr ATARI ST Spezialist in der Schweiz.
0 32 / 25 45 53

Lindenweg 24, 2503 Biel

CH-3084 Bern

Computer Corner
Ihr ATARI ST-Partner in Bern

Hard-Software · Beratung
Midi-Schulung · Service
Seftigenstr. 240 · 3084 Bern-Wabern
Zentrum Wabern · Tel. 031/54 51 00

CH-4054 Basel

Wir sind Amiga- und ST-Freaks.
Unsere Öffnungszeiten:
Von Dienstag bis Freitag
9.30 – 12.30 und 14.00 – 18.30 und
am Samstag 9.30 – 16.00

SYSAG COMPUTERCENTER
Basel: Tel. 061/39 25 25 · Holeestrasse 87 · 4054 Basel
Aarau: Tel. 064/22 63 33 · Kasernenstrasse 26 · 5000 Aarau

CH-4625 Oberbuchsiten

STECTRONIC M. Steck
Electronic-Computer-Shop

Hauptstr. 104/137
CH-4625 OBERBUCHSITEN
Tel. 0 62/63 17 27 + 63 10 27

CH-5430 Wettingen



Zentralstrasse 93 · CH-5430 Wettingen
Tel. 056 / 27 16 60 · Telex 814 193 seco

CH-8001 Zürich

ADAG LASERLADEN
SEILERGRABEN 41 . 8001 ZÜRICH
MO-FR: 12-18.30 TEL. 01/251 49 34
DTP... LASERPRINTS ... ET CETERA!

CH-8006 Zürich

ADAG
COMPUTER-SHOP
UNIVERSITÄTSSTR. 25
8006 ZÜRICH TEL. 01/252 18 68

Computer-Center P. Fisch

Stampfenbachplatz 4
8006 ZÜRICH
01/363 67 67

CH-8021 Zürich



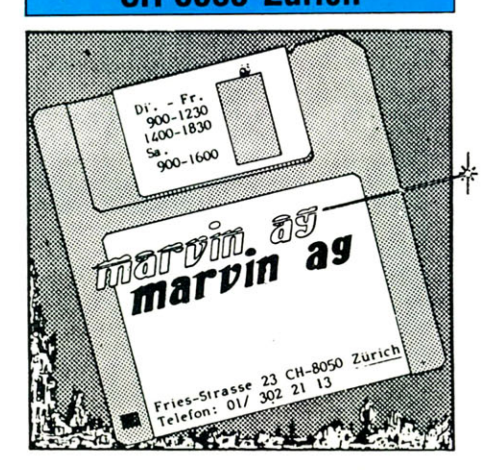
Langstrasse 31
Postfach
CH-8021 Zürich
Tel. 01 / 241 73 73
Telex 814 193 seco

CH-8021 Zürich

VILAN
Das Warenhaus der neuen Ideen

Ihr Computer-Fachhandel an der Bahnhofstrasse 75 · Zürich

CH-8050 Zürich



Kleinanzeigen

BIETE HARDWARE

32-Bit-Userport mit Eprom-
Brenner kpl. m. Kabeln, Gehäuse u. Software evtl. m. best. Eprom-Bank DM 385,– VB. 069/531595

MEGA-ST, Sammelbest. Suche für S.-Best. User, die einen Mega haben wollen. Besonderheit: Er hat ein zweites LW (NEC) einge. **5%–10% Ersparnis!** Schreibt an: Th. Rapp, Curtiusweg 21, 2000 Hamburg 26

SF 314 Tel.: 02241/203983

STAR NL10 m. IBM-Interface und Druckerkabel; 400,– 04183/5494

Monitor SM124, Maus, SF354 neu, Festplatte SH204 neu, Drucker Seikosha SP1000, kompl. 2100 DM Tel. 06724/1304

Hard-Disk 21 MB (Seagate) für ST! **neu,** ohne Gehäuse/Netzteil, sonst kompl., **890,–** 07222/6338

260ST, 3,5" Floppy – 1 MB, ROM-TOS f. Bastler, VH 400,– Tel. 06150-81460

Mega ST2 + SH205 + SM124 + Laser SLM804 mit Garantie
Preis VB 6850,– Tel. 08139/6444
Eventuell Laser alleine

Funkuhr DCF77 T. 07131/161151

Sanyo CD 3185A RGB/PAL Monitor incl. Kabel DM 370 06782-4840

Zusatzplatine für den Anschluß eines Ompti-Controllers an den ATARI ST zu verkaufen. VB 59,– Tel. 0511/469817 oder 320054 14 h

ST Floppy 360 100,– 089/267466

Verkaufe **Festplatte 10 MB** für DM 300,– (07253) 22375 ab 18 Uhr

Scanner f. STAD nur DM 180,–
Schnellste Lieferung.
Tel.: 0208/482249 (nach 18.00 Uhr)

Atari SF354, neuwertig, 150 DM, ab 16.30 Uhr, Tel. 02991/6578

5¼-Zoll-Floppy 1.6 MB brutto 40/80-Track Umschaltung 360 DM
Modem 1200/300 Bd vollduplex HAYES kompatible auto dial& answer 350 DM Tel: 0201/515151

BIETE SOFTWARE

GFA-Basic-Compiler DM 50, Color Star DM 50, Pro-Text DM 90, DB-Master one DM 50, GFA Basic-Interpreter DM 50, Sub-Battle DM 30, alles Orig. 06782-4840

★ Atari **Lernkurs für Anfänger** ★
Disk, Info: B. Biffinger, Dammw.25
★ CH-3903 Naters (Rückporto) ★

Wärmebedarfsberechnung DIN4701
KZahlberechnung DIN4108
Wärmeschutznachweis WäSchV
Dipl. Ing. V. Koch, Am Mehnacker 11
3563 Dautphetal 3, Tel. 064687652

ST-PD-Mailbox 0211-719261 8N1

★ **Super-Lohn-Einkommensteuer**
Jahresausgleich 87 neu m. Datenspeicher, Kundenverw. Formulardruck, Analyse ab 70,– jährl. Aktu (10,–) Info gg. RP PC-Demodisk 10,– H-I-Software H-Ilchen Niederfelderstr. 44, 8072 Manching 08459-1669

■■■■■■■■■■■■■■■■■■■
■ Atari · Schulmeister ST · Atari ■
■ Die komfortable Noten- und ■
■ Klassenverwaltung mit über 40 ■
■ Funktionen/GEM. Editierbare ■
■ Masken und Parameter: ■
■ (Fächer, Unterbereiche, Ge- ■
■ wichtung, Notentyp, Sortier- ■
■ kriterien ect.) 500 Kbyte Ram ■
■ monochrom-Monitor. Version ■
■ für BRD und SCHWEIZ. ■
■ Ausführl. Info/Freiumschlag. ■
■ M. Heber-Knobloch ■
■ Auf der Stelle 27, 7032 Sindelf. ■
■■■■■■■■■■■■■■■■■■■

Vokabelprgm VocMaster +, V:2.5 Grundwortschatz, Synonyms, Gem 49,–, Info: Gratis, Demo: 20,– Reiner Kocher, T: 089/3134946 Caracciolastr. 16, 8 München 45

Wärmebedarf DIN4701 + K-Zahl
★ Heizflächenauslegung ★ Rohrnetz ★ Demodisk 2-seitig für DM 10 Vorkasse von J. Binder Eichendorffstr. 15 · 5030 Hürth

///I Heilpraktiker V1.1 DM 448
Die Anwendung für die Praxis Info u. Demo DM 20,- (0731/4671)
mss masch software service
Irisweg 45 7900 Ulm-16

■ ■ PD-SERVICE ■ ■
■ PD-Angebot dieser Ausgabe ■
■ Einzeldiskette SS **DM 5,00** ■
■ Doppeldiskette DS z. B. 1+2 ■
■ 33+34, 89+90 usw. **DM 7,00** ■
■ Pakete 1–10, 11–20, usw. ■
■ auf je 10 Disk. **DM 40,00** ■
■ auf je 5 D. (DS) **DM 30,00** ■
■ Porto u. Verpack. DM 3,00 ■
■ V.-Scheck o. NN (+DM 3,50) ■
■ N. Twardoch, Gröchteweg 22 ■
■ 4902 Bad Salzuflen 1 ■

Anti-Virus V1.1
Info bei: Marcus Klocke-Sewing Gronauer Str. 19 – 48 Bielefeld 1

> > > **HANDWERKER!!!** < < <
> Rechnungen, Angebote, Lie- <
> ferscheine, Auftragsbestä- <
> tigungen, Mahnungen und <
> Kalkulation: das ist <
> > > **HAROFAKT** < < <

★ ★ **ST-WINDOWMAKER** ★ ★
Windowprogrammierung für jeden leicht gemacht! Preis: 40 DM
Info bei: J. Velmans
Kuyckheide 60, 4174 Issum 2

PD-Grafik + Art-Libs für Degas + STAD Katalog von Frey, Rheinstr. 12A, 6538 Münster-Sarmsheim

PD-Softw. für Erwachsene
Fischer, Pf. 1619, 4830 Gütersloh
Bitte Rückporto; auch Tausch

Atari-Public-Domain ★ 600 Disk's in IBM-Format auf 3½ Zoll ★ Ab 10 Disk's Rabatt ★ Tlf: 02954-1050

Programme PD-Special für Atari-ST. Liste S/88 gegen Freiumschlag (A4) bei M. Römer-Reinl, Rüdesheimer Str. 34, 6200 Wiesbaden

Wettkampfauswertung für Laufveranst., Ergebnislist., **Urkunden** drucken, f. alle STs, **89 DM,** K. Winterstein, 02524/2376 17 Uhr

MS-DOS Emulator PC ditto 179,- über 2000 Prg. bei uns erhältl. und ST-Software ★ Info 17–21 h SMC H. Keseling Fanny-Lewald-Ring 5, 2050 Hmb80, 040-7351931

Biete PD-Software auf 2-seitigen Disketten! AB 4 DM/Disk!
Volker Losch, Am Göhlen 1
2910 Westerstede 2

Easybank 576 KB-Platine, Profimat, Div. Bücher Tel. 06408-7328

PD-Software ab 2,– DM/Diskette Angebot wie PD dieser Ausgabe Tel: 02721/2432 von 12–21 Uhr

■■■■ **PD ab DM 3,50** ■■■■
■ **PD 1 bis PD dieser Ausgabe** ■
■ **INFO-Liste gegen DM 0,80** ■
■ **G. Grasberger, C-Orff-Str. 9** ■
■■■ **8221 Traunwalchen** ■■■

Angebot wie PD dieses Heftes & eigene PD **ab 2 DM,** 10er-Pakete ab 35 DM, **INFO:** Witaseck, Josef-Neuberger-Str. 26, 4000 Düsseldorf

!!!!!AUFMASS AUF DEM ST!!!!!
! Mit **HAROMASS** erstellen !
! Handwerker ihr Aufmass auf !
! dem Computer! SUPER!!! !
! DM 698,– **DEMO: 20,– DM** s!
! Telefon: 07346/3845 !

Drucker MT90 750 DM; Floppy SF354; **Div. Originale:** Lattice C, HackerII, MS-DOS, GFA-Starter, Dizzy-Wizzard, Video-Construction-Set. Auch Tausch möglich. **07723/4471**

Lohnsteuerjahresausgleich/Einkommensteuer 87/88. Grunddaten + Ergebnis speichern. WasWäreWenn Lohn- u. ESt-Tabellen 1987/88
30 DM Vork. J. Höfer, Grunewald 2a, 5272 Wipperfürth, 02192/3368

Videofilmprogramm unter STbase mit beigefügtem Run-Time-Interpreter. **Schreibe auch Programme zu STbase nach Ihren Wünschen.** Uwe Bremer, Tel. 0211/419240

ST-COMPUTER komplett bis auf 3/86 u. **orig. DATAMAT** 046159389

Ahnenverwaltung m. Listen u. Grafikausg. v. Vor- u. Nachfahren Pfeiffer, Espen-5, 4047 Dormagen

Verkaufe Originalprogramme
STEVE-Programm 250,–
GFA-Basic V2.0 80,–
Tel: 09392/7874

■■ **Brandneue PD-Software** ■■
■ Fast täglich neue Programme, ■
■ direkt vom Autor, die es nur ■
■ bei mir gibt! Riesiges Ange- ■
■ bot auf über 30 Seiten! **Alle** ■
■ **Programme sind einzeln aus-** ■
■ **wählbar!!!** Der absolute Hit: ■
■ NUR 1,5 PF PRO KBYTE ■
■ Sie können **eigene** Disketten ■
■ senden oder auf **Markendisks** ■
■ bestellen!! Jedem PD-Katalog ■
■ liegt ein **Extrablatt** mit den ■
■ allerneusten Programmen, die ■
■ ständig erfaßt werden, bei!! ■
■ Bearbeitung erfolgt selbst- ■
■ verständlich **postwendend!!!!** ■
GRATISkatalog bei Arne Zingel, Vermehrenring 11a, 24 Lübeck 1

★ ★ ★ HANDWERKER ★ ★ ★
★ ∞∞☎ ist die perfekte ★
★ Fakturierung für Handwerker ★
★ Kinderleicht! Mausgesteuert ★
★ DM 698,– **DEMO: 20,– DM** ★
★ Telefon: 07346/3845 ★

■■■■ **Supergünstig** ■■■■
■ PD aus ST incl. Disk DM 4,70 ■
■ Versand Schneider 0281/61772 ■

Original! 32-Forth, Starglider*, XTRON, Tracker*, Time-Bandits* **je 35 DM** (*engl. Anl.)
Tel. 07173/8009

★ **PD-AKTUELL,** das Infomagazin für den PD-Anwender. Probeheft von **EUSAX PD-Service,** Haus Gravener Str. 85, 4018 Langenfeld

□ □ **SUCHEN SIE PD-SOFT?**
□ □ Dann fordern Sie unseren
□□ neuen Gratiskatalog
□ □ **Klaus Kohler**
□ □ Don-Carlos-Str. 33 B
7 Stuttgart 80

Wordplus-Druckertreiber f. **alle STAR NL-10 u. LC-10.** Kompl. angepaßt, volle Befehlsausnutz. m. ca. 100 KB Anleit.; HEX+CFG. Grundl Treiber v. K. Plüher. 30,– p. Rech. **Wörterbuch 240 KB:** 10,– DM incl. Disk. U. Köhler, Mont-Cenis 537, 4690 Herne 1 – Tel. 02323/64315

■■■ **!! PD jetzt DM 5,50 !!** ■■■
■ PD von 1 – diese ST-Ausgabe ■
■ **auf Mark.-Disk SS je DM 5,50,** ■
■ P&V. DM 5,00, ab 20 St. frei! ■
■ V.-Scheck od. NN (+ DM 1,70) ■
■■ NEU! ■ NEU! ■ NEU! ■■
■ **Weg mit dem PD-Müll** ■
■ **Qualität vor Quantität!!!** ■
■ **Jetzt eigene PD-Sammlung** mit ■
■ Bereichssortierung, **als Disk** ■
■ **o. einzel. PRG's** nach Kbyte! ■
■ **Viele Sonderangebote!!!** ■
■ 2,50 DM-PD-Disk, Gratis-PD..! ■
■ **SUPER GRATIS-KATALOG!** ■
■ J. Rangnow · 7519 Eppingen ■
■ Talstr. 8 · 07262/5131 ab 17°° ■
■■■ **PD-Express-Versand** ■■■

Public-Domain-Blitzversand!
Riesenauswahl! Preise:
ab DM 4,– einseitige Disketten
ab DM 6,– doppelseitige
Inclusive Diskette!!!
Auch alle ST-Disks! Auf Doppeldisks beliebig kombinierbar!
Gratisliste anfordern bei:
A. Gauger Software
Buhlstraße 16a, 7505 Ettlingen
07243/31828
Bitte Computertyp angeben!!

Statik- u. Matheprogr. unter Gem
Demo-Disk u. ausf. Info (10 DM bar) V. Scheck) von: Dipl. Ing. Jürgen Bullmann, Zur Hindenburgschleuse 3 3000 Hannover 71

★ ★ **iks Public Domain** ★ ★
bitte beachten Sie unsere Anzeige an anderer Stelle in dieser Ausgabe. Es lohnt sich!

Von Bauleiter für Bauleiter!!
Aufmass-Massen-Abrechnungs-Prg
f. ATARI-ST 245,– T. 0821-499518

★ Atari **Lernkurs für Anfänger** ★
★ Disk, Info: B. Biffiger, Dammw. 25
★ CH-3904 Naters (Rückporto) ★ ★

★ ★ NEUE DIMENSIONEN ★ ★
PD-Software für Emulatoren auf Atari ST: **Aladin, Supercharger, PC-Ditto, CP/M, Amiga u.v.m.!** Info: DM 0,80 – Carsten & Marcus Postfach 65 06 02, 2000 Hamburg 65

BERLIN: PD-Software, 200 Disks z. Mitnehmen je DM 7,–, auch Großauswahl an Kaufsoftware und Zubehör. **Computershop Behrendt,** Fürbringerstr. 26, 1–61, Tel. 6917666 BTX 0306917666

Von Lehrer für Lehrer! Notenverw.prg **Markbook plus** in GEM. Über 20 versch. Funkt. f. alle gäng. Notensyst. Spez. vers. auf Wunsch; Info gegen Freiumschlag.
H. J. Merkel/Nahestr. 28/6600 SBR

ATARI ST-LOTTOSTATISTIK 39 DM, 20 seit. Dokumentation vorab gegen 2,40 in Briefmarken. F. Klein, Zur Fröhn 15, 6601 Heusweiler, Tel. 06806/8901

SPORT-MANAGER-ST!
Professionelle Sportverwaltung für ST! Alles für den gut organisierten Wettbewerb. Preis: 130,– DM, Scheck oder NN bei: MTC, Postfach 56, A-6027 Innsbruck. Info gegen Rückumschlag!

SUCHE HARDWARE

Verkaufe Schneider DMP 3000 Drucker Epson + IBM kompatibel DM 300, und einen Centronic GLP Drucker DM 100, Tel. 07564/2747

Suche SH 204/5 02273/1472

SUCHE SOFTWARE

Suche **Astrologie u. Bio.** Prg.
L. Barnhofer, Postf. 1403, 523 Altenk.

Suche 6802/6805-Crossassembler für ATARI 1040/ST nach 20:00
Tel.: 06128/42600

TAUSCH

Tausche neuste Software Listen an Wolfgang Müller, Landsberger Str. 93 8901 Königsbrunn

Tausche PD-Software (auch gegen Leerdisks!) Für Liste bitte 1 Disk + Rückporto an: D. Metz; Bahnhofstr. 84a; 2153 Neu Wulmstorf

KONTAKTE

Programmierer, nebenberuflich für LATTICE C gesucht 044032167

Suche Kontakte im Raum 89 zum Training mit ADIMENS-ST u. and. Progr. u. Tausch. Tel. 08233/20769

VERSCHIEDENES

Platinen-Layouts!
von Elektroniksch., **Atari-ST Plotterservice** für alle CAD Progr., 0431-180975
U. Tams, Westring 273, 2300 Kiel

ST-Computergrafik-Gestalten.
Clip-Art Disk ü. 1000 Zeichen
Scanner-Service-HAWK PD-Grafik
PRG.-Bild. Software Grafik Info
R. Probst Produktdesigner, grad.
2121 Dahlenburg OT Ellringen 12

Adimens-ST-1.6-Handbuch gesucht: W. Klemke, Am Krausen Baum 6 4000 Düsseldorf 31

★ **Neue Fonts für Signum** ★
★ Professional Quality ★
★ für 9-, 24- und Laderdrucker ★
★ H. Manschus, Tel. 0651/76814 ★
★ Karl-Marx-Str. 70b · 55 Trier ★

Übernehme Programmieraufgaben in GFA BASIC
P. Neurohr, Eschersheimer-Landstr. 273, Tel: 069/567298

STARKE SOFTWARE



INTEGRIERTES SOFTWAREPAKET FÜR DEN KLEIN- UND MITTELSTÄNDISCHEN HANDEL.

Das Programm mit der hohen Geschwindigkeit und Bedienerfreundlichkeit.

Leistungsmerkmale
- Verwaltung von Artikeln, Kunden, Interessenten, Lieferanten
- Erstellung von Adress-, Inventur-, Provisions- und Postenlisten
- Datenkapazität nur von externem Speichermedium abhängig
- Erstellung von Bestellungen, Angeboten, Auftr.-Bestätigungen, Lieferscheinen, Rechnungen, und Gutschriften
- Automatisch Bestandsüberwachung
- Automatisches Mahnwesen
- Einfache Weiterverarbeitung von bereits erstellten Dokumenten
- MWSt-Änderung für alle Artikel mit einem(!) Befehl
- 4 VK-Preise pro Artikel, jeder Artikel einzeln rabattierbar
- 1 Zusatz-Preisfeld pro Artikel
- Kunden- und Artikelumsatzstatistik
- Komfortable Druckeranpassung
- Schnittstelle zu 1st Word (Plus)
- Verarbeitung von 1st Word-Texten (Plus)
- Serienbriefunktion, Etiketten
- Druckermasken für alle Dokumente frei erstellbar, somit Anpassung an Vordrucke aller Art
- einfache Benutzerführung, hohe Bedienungsfreundlichkeit
- kompakter Programmcode (nur 80 KB), dadurch hohe Verarbeitungsgeschwindigkeit
- ständige Programmpflege & UP-Date-Service
- Handbuch

DM 899,–



DIE KOMFORTABLE BUCHFÜHRUNG FÜR FREIBERUFLER UND KLEINGEWERBETREIBENDE.

Ohne Vorkenntnisse leicht zu bedienen.

Leistungsmerkmale
- Freie Wahl des Abschlußzeitraums (Periode)
- Voreinstellung von bis zu 5 Steuersätzen
- Einrichtung von bis zu 10 laufenden Konten
- Einrichtung von bis zu 200 Einnahme/Ausgabe-Konten
- Erfassung von bis zu 2.500 Buchungen pro Periode
- Volle Eingabekontrolle auf Fehler (Löschen der letzten Buchung)
- Erstellung von Bilanzen
- Erstellung von Journalen
- Erstellung von Saldenlisten und Kontenrahmen
- Datenaufbereitung für USt.-Voranmeldung
- Ausdruck aller wichtigen Daten
- Anpassung an jeden Drucker
- Schnittstelle zu anderen Programmen
- Periodenabschluß mit automatischer Saldenübertragung in die Datei für die Folgeperiode
- Voll in GEM eingebunden
- Einsetzbar auf allen ATARI ST und MEGA ST mit S/W und Farbmonitor
- UP-DATE-SERVICE
- Handbuch

DM 149,–



ORDNUNG, ÜBERBLICK, PLANUNG – MIT DER HAUSHALTSBUCHFÜHRUNG FÜR ALLE.

Leistungsmerkmale
- Monats- und Jahresabschluß
- Benenung von bis zu 50 laufenden und 200 Kostenkonten
- Verwaltung von bis zu 50 Dauerbuchungen mit wählbarer Frequenz
- Bis zu 2.500 Buchungen pro Periode
- Volle Eingabekontrolle auf Fehler
- Erstellung von Bilanzen, Journals, Saldenlisten und Kontenrahmen
- Ausgabe von Buchungslisten mit Korrekturmöglichkeit
- Saldenbildung von Kontenblöcken
- Anpassung an jeden Drucker
- Schnittstelle zu anderen Programmen
- Voll in GEM eingebunden
- UP-DATE-SERVICE
- Handbuch

DM 129,–

alle Preise sind unverbindlich empfohlene Verkaufspreise

Heim Verlag
Heidelberger Landstr. 194
6100 Darmstadt-Eberstadt
Telefon 06151-56057

BESTELLCOUPON

an Heim Verlag
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt

Bitte senden Sie mir
____ Stück **TKC-FAKTURA ST** z. Preis v. DM 899,–
____ Stück **TKC-EINNAHME/ÜBERSCHUSS ST** z. Preis v. DM 149,–
____ Stück **TKC HAUSHALT ST** z. Preis v. DM 129,–
zuzüglich DM 5,– Versandkosten unabhängig von der Bestellmenge.
☐ per Nachnahme ☐ Scheck liegt bei ☐ per Vorausberweisung

Name, Vorname ____
Straße, Nr. ____
PLZ, Ort ____
Unterschrift ____
Benutzen Sie auch die im ST-COMPUTER vorhandene Bestellkarte

G-Data Produkte

G-Ramdisk II	48,--
G-Diskmon II	98,--
Harddisk-Help & Extens.	129,--
Interprint II	49,--
Interprint II Ramdisk	99,--
As Sound Sampler II (Softw.)	149,--
Disk Help	79,--

Omicron-Produkte

Omicron-Basic (Disk.)	179,--
Omicron-Basic (Modul)	229,--
Omicron Compiler	179,--
Omicron Assembler	99,--

Royal-Produkte

Disk-Royal (Disk-Monit.)	89,--
Katpro-Royal (Disk.-verwalt.)	89,--
Wizard-Royal (Spiel)	139,--
Voc-Royal (Vokabeltrainer)	79,--

Desktop Publishing

Calamus	998,--
Publishing Partner	498,--
Fleet Street Editor	348,--
Timeworks Publisher	389,--

Public-Domain-Service

Wir liefern auf erstklassigem Diskettenmaterial die PD-Software aus ST (siehe PD-Liste in dieser Ausgabe) und **1000-1042** (siehe unten; nur bei uns erhältlich)! **jede Diskette nur 8,--**

1000. Diskettenverwaltungspr.
1001. Digimusik Oxygen
1002. TOS vom 2.6.86
1003. Basicpr., Terminalpr.
1004. Pyramide, ST Pic Zeichenprogramm
1005. Finanzpr., Musikpr.
1006. Filecopy, Convertierpr. für Monitor
1007. Diskmonitor
1008. Fonts, Kermit GEM
1009. Sounds, Analoguhr, Calc
1010. Dirprint, Dis... Ramcopy
...eiber 1st Word
... Diskmonitor,
1015. Mauseditor, ...
1017. Bilder, Druc... Star
...ter Demo
...gramme
1021. Schach, Com...
... ST Grafik
1023. Mauseditor, ...
...ramdisk,
1026. Druckertreib...
...ext
1028. Moire-Bilder...
...essories
1030. Grafikdemo
...tei, Grafikdemo
1033. Forth Interpreter, Logo
Kniffel-Spiel
Druckertreiber 1st Word
1034. Basic-Files, Megaramd.
1035. Source-Code-Lister, Sound-Demos
1036. Degas-Files, Degasfonts
1037. GEM-Demo, Desk-Uhr
1038. Label-Maker, Datetime
1039. Ramdisks, Copy-Pr., Printhelper EPSON
1040. Ramdisk, Spiele
1041. Basic-Utilities, PrintDir
1042. Proff

Neu! MS-DOS Freesoftware

PC-Ditto und Supercharger ermöglichen es MS-DOS Software auf Ihrem Atari ST laufen zu lassen. Mehr als 700 Disketten mit über 10000 Programmen sofort lieferbar! Fordern Sie unsere **MS-DOS PD-Liste** an! Bitte schriftlich mit Angabe des Computertyps.

10er Blöcke: 10 Programme auf jeweils 5 Disketten 1DD (1MB)!! 1-10, 11-20, 21-30, 31-40, 41-50, 51-60, 61-70, 71-80, 81-90 und 101-110, 111-120, 1001-1010 usw... **je Block nur 40,--**

Monatsausgaben ST	**28,--**
Updates (U1, U2, U3, U4)	**8,--**

Bei Best. Computertyp angeben. Liste anfordern!

Atari-Schaltpläne

260 ST / 520 ST	29.80
520 ST+ / 520 STM	29.80
1040 STF	29.80
1040 Erweiterung	29.80
SF 314 / SF 354	je 19.80
SNM 804 / 1050	je 19.80
600 XL / 800 XL	je 19.80
SC 1224 / SM 124	je 19.80

Mouse Pad Super !!
Gleitmatte f. Maus **DM 19,80**

Kuma Software

K-Spell	118,--
K-Seka (Assembler)	198,--
K-Spread 2 (Tabellenkalk.)	198,--
K-Graph 2 (Grafik)	148,--
K-Comm (Kommunikat.)	148,--
K-Resource (Editor)	118,--
K-Word 2 (Textverarb.	118,--
K-RAM	89,–
K-Switch (Utility-Prg.)	99,–
K-Minstrel	89,–
dB man 4.0 (neu)	624,--

Neu • Neu •  Software

PC DITTO (Softwareemulation)

Nun können Sie Tausende von MS-DOS und PC-DOS-Programme auf Ihrem Atari ST ablaufen lassen!

- Verwandelt Ihren Atari ST in einen IBM-Klon
- Für FARB- und MONOCHROM-Monitore
- Stützt auch Festplatten
- Stützt Seriell / Parallel angeschlossene Drucker
- Kompatibel mit europäischer Tastatur
- Leicht verständliches, deutsches Manual

(Testbericht in Atari ST 12/87) **178,--**

Tommy Software

1ST Speeder **89,–**
Beschleunigt das Arbeiten mit Diskette bis zu Faktor 10!

MusiX32 **89,–**
Der Musikstandard auf dem ST

1ST Freezer **148,–**
Ermöglicht komprimierte Speicherdumps auf Disk.

Dizzy Wizard **69,–**
Tolles Spiel • S/W und Farbe

STandard BASE III (Knupe)
Professionelle Datenbanksprache!
Voll kompatibel zum Weltstandard dBASE III • Inkl. 600 Seiten deutschem Handbuch • **Hardware-Anforderungen:** Atari ST mit 1MB RAM u. ROM-TOS, Monochrom-Monitor, Diskettenlaufwerk SF 314. **698,--**

Supercharger **698,--**
MS-DOS-Hardwareemulator
Erweitert Ihren Atari ST um einen IBM-XT (oder AT):
- volle PC-Kompatibilität
- sehr schneller Datentransfer
- eigenständiger 8086 Prozessor
- Anschluß an DMA-Port mit spez. Interface
- Erweiterungsmöglichkeiten (80286)

TEMPUS Editor 2.0 (CCD)
Hohe Verarbeitungsgeschwindigkeit • Hoher Programmkomfort • **TEMPUS** hilft dem Hobby- u. dem professionellen Programmierer, kostbare Zeit einzusparen. **109,--**

TIM 1.1 (Time is Money)
Buchführung für Atari ST (Monochrom)
(Test Atari ST 7/87) **298,--**

Stets aktuell  Software

Textverarbeitung:

1st Word /Word Plus	99,–/199,--
1st Proportional	148,--
Wordstar / Mailmerge	199,--
Textomat ST	99,--
Protext ST	148,--
Becker Text ST	199,--
Megafont ST	119,--
Word Perfect	910,--
Typsetter Elite	139,--

Grafik • CAD:

Art Direktor	173,--
Film Direktor	188,--
Monostar / Colorstar je	78,--
Maps & Legends	98,--
Degas	169,--
MICA	298,--
Animator	119,--

Campus Maschinen- und Werkzeugbau, Fertigungs- und Elektrotechnik sowie Architektur, einfach alles, was mit technischem Zeichen zu tun hat, meistert **Campus** professionell. (Demo 20,--) **998,--**

Programmiersprachen:

MCC Lattice C-Compiler	298,--
MCC Pascal Compiler	248,--
MCC Macro Assembler	168,--
MCC Make	168,--
MCC Lisp	448,--
MCC BCPL-Compiler	329,--
Pro Fortran	448,--
Pro Pascal	448,--
Modula II (Standard)	298,--
Modula II (Develop.)	448,--

Datenbanken:

Adimen ST neu	199,--
Aditalk	189,--
DB-Man deutsch neu	395,--
Logistix	398,--
Profimat ST	99,--
dBase II ST deutsch	348,--

Side Click	145,--
Copystar V2.2	169,--

Steinberg Musikprogramm:

Twenty Four	478,--

HEIM PRODUKTE

Bücher: Omicron Basic	29,--
Progr. in Omicron Basic	49,--
Das große VIP -Buch	49,--
C auf dem Atari ST	49,--

Software: Salix Prolog	198,--
Steuer Tax ´87 Vers. 2.7	98,--
Vers. 3.7	159,--
ST Aktie	69,--
ST Plot	59,--
ST Print	59,--

Aladin (Mac Emulator)

Die Chance für Sie, auf Ihrem Atari ST Macintosh-Software zu fahren!
- höhere Bildschirmauflösung
- höhere Verarbeitungsgeschwindigk.
- integrierte, resetsichere Ramdisk
- Hard- und Software incl. Handbuch

398,--

Mac-Betriebssystem (ROM-Satz) **195,--**
Info anfordern! (schriftl.)

Zubehör

Etiketten endl., 70x70 (200 Stk)	16,–
10 Disketten 1DD	39,–
10 Disketten 2DD	49,–
Diskettenbox 40/80	39,–/46,–
Druckerkabel	38,--
Floppystecker	9.80
Monitorstecker	6.80
Monitorbuchse	14.80.
PAL-Interface III Anschluß Ihres ST's an Farbfernseher	198,--
Abdeckhauben:	
Konsole 520/1040	29,–
Monitor (124/1224)	39,–
Farbbänder a. A.	
CSF-Gehäuse für 260/520	148,--
ROM-Satz für alle ST's	148,--
Akustikkoppler 300/1200	378,--
Literatur a.A.	
THINGI-Konzepthalter	24,80
Trackball von Atari	98,--
Monitor-Box umschaltbar Monochrom / Color	58,--
Computertisch grau, eckig	298,--
hochwertige Qualität rund	348,--

Alles aus einer Hand !

Wir unterhalten ein reiches Lager! Alle Atari Soft- und Hardwareprodukte stehen vorführbereit in unseren Ausstellungsräumen. Besuchen Sie uns doch einmal.

Weide-Produkte

Echtzeituhr 129,-- Jede Uhr im Rechner getestet und gestellt. Interner Einbau ohne Löten. 10 Jahre Laufzeit.

Speichererweiterung 298,-- Interner Einbau ohne Löten. Gut bebilderte Einbauanleit. Erw. auf 1MB für 260/520 STM.

Video Sound Box 248,-- Klangkräftige 3-wege Box mit integriertem HF-Modulator zum Direktanschluß aller Atari ST an den Fernseher. Unübertroffene Bildqualität!

Karl-Heinz • Potsdamer Ring 10 • 7150 Backnang
Telex 724410 weeba d • Kreissparkasse Backnang
(BLZ 60250020)74397 • Postgiro Stgt. 83326-707

weeske
COMPUTER-ELEKTRONIK

gültig April 88

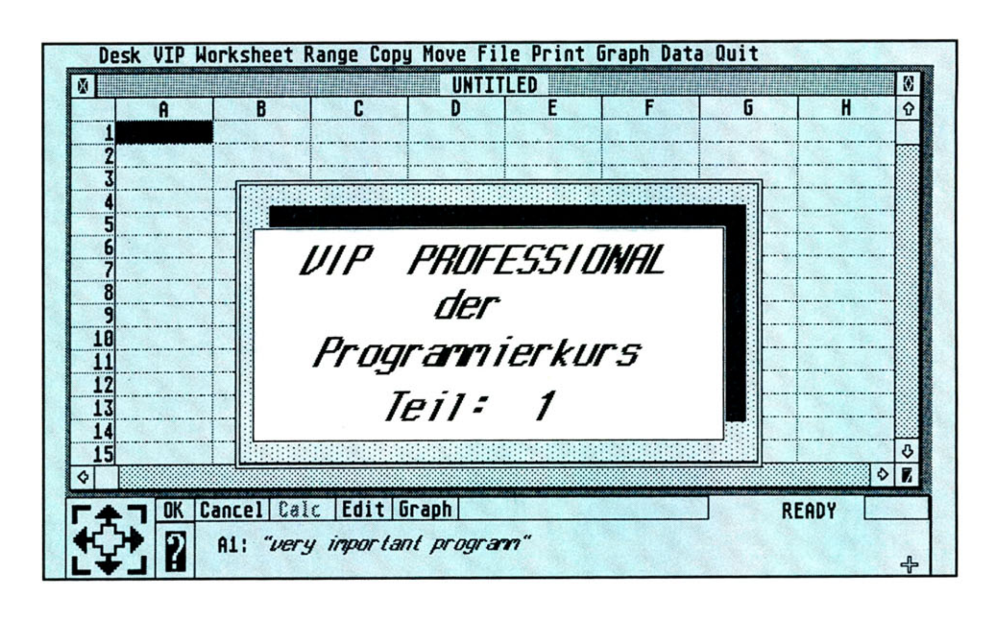
Zahlung per Nachnahme oder Vorauskasse (Ausland per Scheck).
Versandkostenpauschale (Inland 6,80 DM/Ausland 16,80 DM).
Bitte bei Bestellung + Infoanforderung den Computertyp angeben!

07191/1528-29 od. 60076

BITTE EINSTEIGEN

DER V.I.P.-KURS FÄHRT AB

Tabellenkalkulationsprogramme für den ATARI ST erfreuen sich nach wie vor großer Beliebtheit, für manchen Anwender geht kein Weg an solch einem Rechen-, Datenbank- und Grafikprogramm vorbei. Die sogenannten integrierten Programmpakete sind jedoch durch ihren sehr großen Befehlsumfang nicht immer leicht zu bedienen. Unser Kurs soll den NEULING oder "absolute Beginner" (David Bowie möge es mir verzeihen) auf dem Gebiet der Tabellenkalkulation, Schritt für Schritt an die Bedienung von VIP PROFESSIONAL, einem LOTUS 1-2-3 ähnlichen Tabellenkalkulationsprogramm, heranführen.



Was'n V.I.P ?

Der Kenner würde VIP mit "very important programm" übersetzen. Unser Vip-Professional wird in Deutschland von der Computertechnik Kiekbusch GmbH seit 1984 auf dem deutschsprachigen Markt angeboten. Seit der Programmfreigabe ließen sich immerhin 4500 Pakete dieses Tausendsassas veräußern, eine wohl unbekannte Menge zusätzlicher gravitationsloser Freundschafts-Sicherheitskopien wird die Zahl der Anwender wohl auf über 10000 schnellen lassen.

Gut finde ich, daß nur registrierte Anwender von dem reibungslos laufenden Update-Service der Kiekbusch-Mannen Gebrauch machen können, denn inzwischen kann man das Programm mit der Freigabe 1.4 in zwei Programmversionen erwerben oder updaten lassen.

Die TEXT-VERSION, auch Turbo-Version genannt, zeichnet sich durch (MS-DOS-Anwender bitte überlest diesen Absatz) relativ hohe Verarbeitungsgeschwindigkeit gegenüber der GEM-Version aus. Die GEM-Version (das ist die, mit der man die Mäuse hüpfen lassen kann) hat leider nur das GEM aufgesetzt bekommen und ist nicht, wie es sich heutzutage gehört, völlig dem GEM-Zauber verfallen.

Das Vip Professional kann seine Verwandschaft mit dem Top-Renner der Tabellenkalkulationsprogramme, dem LOTUS 1-2-3, nicht verschweigen. Mit den ersten Versionen von 1-2-3 setzte man mit der Einführung von macro-programmierbaren Tabellenkalkulationen einen neuen Trend. LOTUS 1-2-3 wurde allerdings ausschließlich für die MS-DOS-WELT geschaffen und befindet sich heute schon in der 3.Generation.

Die Atari-Anwender schauten recht lange neidvoll auf dieses 1-2-3, so daß sich die VIP TECHNOLOGIES in Santa Barbara, Californien, an die Entwicklung eines Clone-Ablegers von 1-2-3, ausschließlich für die 68000-Welt machten. Das gelang den Entwicklern nach einigen Anlaufschwierigkeiten recht gut und nach nunmehr 3 Jahren seit der Programmeinführung erhält man ein fast fehlerfreies Programm, das es einfach in sich hat.

Anfänger aufgepaßt !

Den Fans von alten Heinz-Rühmann-Filmen empfehle ich, die "Feuerzangenbowle" nochmals Revue passieren zu lassen, sagte da nicht ein Professor: "Soo, nu stelle MIR uns mal janz doof". Doch nun genug mit diesem Smalltalk, fangen wir an!

Am besten läßt sich eine Tabellenkalkulation, auch Rechenblatt genannt, mit einem Blatt Papier vergleichen. Teilen Sie Ihr Blattin Spalten und Reihen auf.

Beschriften Sie die Spalten mit Buchstaben von A bis H, die Reihen numerieren Sie mit Ziffern von 1 bis 19. Wenn wir nun die Spalten und

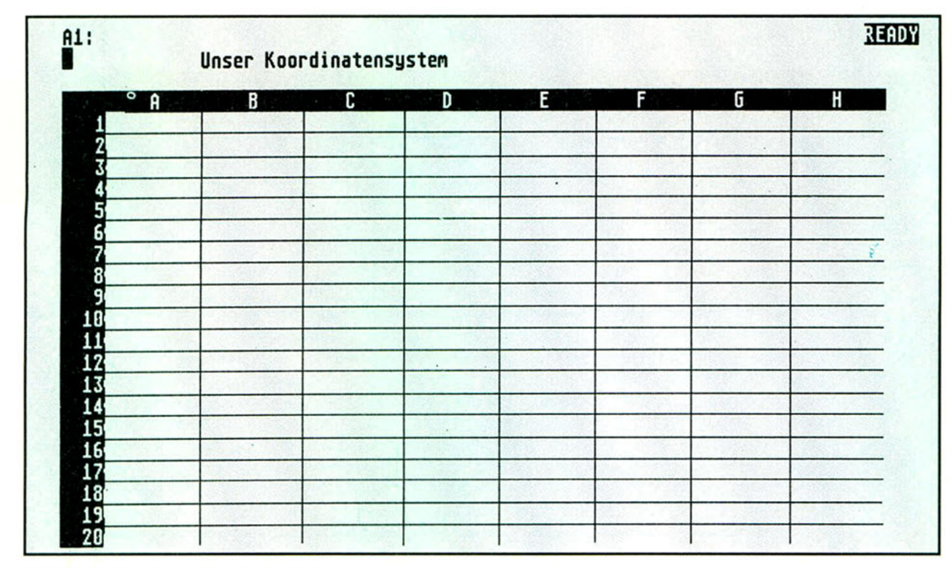


Bild 1: Leeres Koordinatensystem

Reihen durch Linien trennen, erhalten wir ein Koordinatensystem mit Zellen. Diese Zellen werden durch ihren Platz im Koordinatensystem unseres Blattes mit den Koordinatenschnittpunkt benannt.
Also nochmal ganz langsam: die Zelle im linken oberen Eck erhält den Namen A1 die Zelle im linken unteren Eck den Namen H19 und so weiter, und so weiter......
Eine Zelle hat also immer einen festen Namen in unserem Rechenblatt, zusätzlich kann sie unterschiedliche Attribute erhalten, die sich aus Zellenname, Zellenbreite, Text- oder Labelvorgabe, Schutz-Bemerkung und so weiter zusammensetzen können. In unserem Bildchen sehen wir die Zelle C6 mit einer Zellenbreite von 9 Stellen, wir haben nur Texteingaben für diese Zelle vorgesehen und haben den Zellen-Schutz mit U (für unprotected = ungeschützt) vereinbart. Das war jetzt ein wenig kompliziert, aber dazu noch später.

Hurra, unser Baby lernt laufen !

Ich gehe davon aus, daß Ihnen das Papierkritzeln zu langweilig geworden ist und Sie Ihr Vip-Professional bereits geladen haben.
Das leere Arbeitsblatt enthält in der TEXT-Version keine Trennlinien von Zelle zu Zelle, GEM-Vips können mit GRID Ihr Gitter zu- oder abschalten.
Doch es dürfte auch dem Laien nicht schwerfallen, sich solch ein Koordinatensystem sinnbildlich dazuzudenken. Der schlaue Anwender wird schnell feststellen, daß mit einem Arbeitsblatt dieser abgebildeten Größe nicht viel anzufangen ist. Begeben wir uns doch auf die Reise, lernen wir die Grenzen des VIP-Universums kennen.
Versuchen Sie mit der CURSORTASTE (SÜD) über die 20 Reihen-Grenze hinauszupreschen. Siehe da, unser Blatt wird größer. Doch jetzt runter von der Cursortaste, unser ATARI puffert nämlich bei gedrückter Taste jeden Intervallanschlag, und es kann passieren, daß die Bildschirm-Expedition zu einem endlosen Marathonlauf ausartet.
Unser 2. Versuch soll uns nun seitenweise an das Reich der Mitte heranführen. Halten Sie die SHIFT-Taste und die CURSORTASTE (SÜD) gemeinsam einmal gedrückt. Wir stellen fest, unser Marsch wird nicht so anstrengend.
Im 3. Versuch hören wir auf zu marschieren und nehmen das Flugzeug. Drücken Sie bitte die INSERT-Taste; auf der rechten Bildschirmunterseite erscheint das Wörtchen “END”, danach die CURSORTASTE (SÜD) und siehe da - wir haben die Grenze unseres Universums schon erreicht. Immerhin liefert unser Blatt 8192 Zeilen !
Doch wie sieht’s denn mit unserer Orientierung nach “rechts” aus, also in Richtung der Buchstaben X,Y,Z ?
Die Tastenbedienung haben wir ja mit unterschiedlichen Kombinationen schon kennengelernt, auch hier nehmen wir das Flugzeug: mit der Taste INSERT (END-Anweisung) und der CURSORTASTE (OST) fliegen wir gen Osten.
Bis zur Spalte IV reicht unser Sprit, also 256 Spalten ! Lassen wir uns nicht von dieser Doppelbuchstaben-Spaltenbezeichnung (uhh, langes Wort) verwirren, denn Vip zählt seine Spalten wie folgt:
X, Y, Z, AA, AB,......IV (Ganz einfach oder ?) Ich empfehle Ihnen dringend etwas Bewegung, damit meine ich natürlich Bildschirmbewegung in unserem Vip. Bei großen Anwendungen sind die nun mal eingeprägten Tastenkombinationen sehr hilfreich.
Hier nochmal die Zusammenfassung unserer Trimm-Dich-Aktion:

A1: READY
Was ist eine Zelle ?
C6 9 T U Zelle C6

Bild 2:Koordinatensystem mit Zelle C6

a) reihen-/spaltenweises Fortbewegen
= CURSORTASTEN

b) seitenweises Fortbewegen
= SHIFT + CURSORTASTE

c) Sprung an den Rand
= INSERT-TASTE, CURSORTASTE

d) Sprung auf A1
= CLRHOME-TASTE

So, nun müssen wir uns aber beeilen, unser Atari läuft so langsam aber sicher warm, und wir haben noch nichts Produktives erreicht. Tragen Sie in Ihr Vip-Blatt nun in der Zelle A1 den Wert 100, in Zelle A2 den Wert 230, in Zelle A3 den Wert 400, in Zelle A4 den Wert 400, in Zelle A5 den Wert 900 ein.
Man kann diese Eintragung auf zwei Arten durchführen:

1.) Wert eingeben, mit Return bestätigen, CURSORTASTE (SÜD) zur nächsten Zelle.

oder

2.) Wert eingeben, CURSOR TASTE (SÜD) drücken und schon sind wir in der nächsten Zelle.

Sie sehen VIP interpretiert im Eintragungsmodus für Zellen auch die CURSORTASTEN als RETURN (Enter)+Richtung.
Leider müssen wir bei der Addition unserer eingetragenen Zellenwerte unser Gehirn anstrengen, bis wir in der Zelle A6 die Summe 2030 eintragen können.
Damit unser Gehirnschmalz nicht zum Sieden kommt, ließe sich nun im Rechenblatt in der Zelle A6 ein Befehl zur Summenbildung eintragen. Hierfür gibt es mehrere Möglichkeiten:

a) Schreibe in die Zelle *A6* folgende Befehlsfolge:
+A1+A2+A3+A4+A5(Return)
Wir teilen unserer Zelle A6 mit, daß in ihr die Zellen A1, A2, A3, A4 und A5 miteinander addiert und mit dem Ergebnis am Bildschirm angezeigt werden sollen.

b) Wesentlich eleganter kann man diese einfache Addition mit dem Summenbefehl des Vip-Sprachschatzes durchführen.
Schreibe in die Zelle *A6* folgende Befehlsfolge:
@SUM(A1..A5) (Return)
Sie sehen, auch mit der Formel *@SUM(Bereich)* lassen sich Additionen durchführen.
Bei großen Bereichen empfiehlt sich als oder Summenbefehl schon von selbst.

c) Die 3. Variante der Zellenbereichserfassung ist das "Lassowerfen" oder "Einkreisen", wie es die alten Trapper unter den Vips bezeichnen.
Stellen Sie mit den CURSORTASTEN Ihren Zellenzeiger wieder auf Zelle A6 und führen die Eingabe nach Tabelle 1 durch:

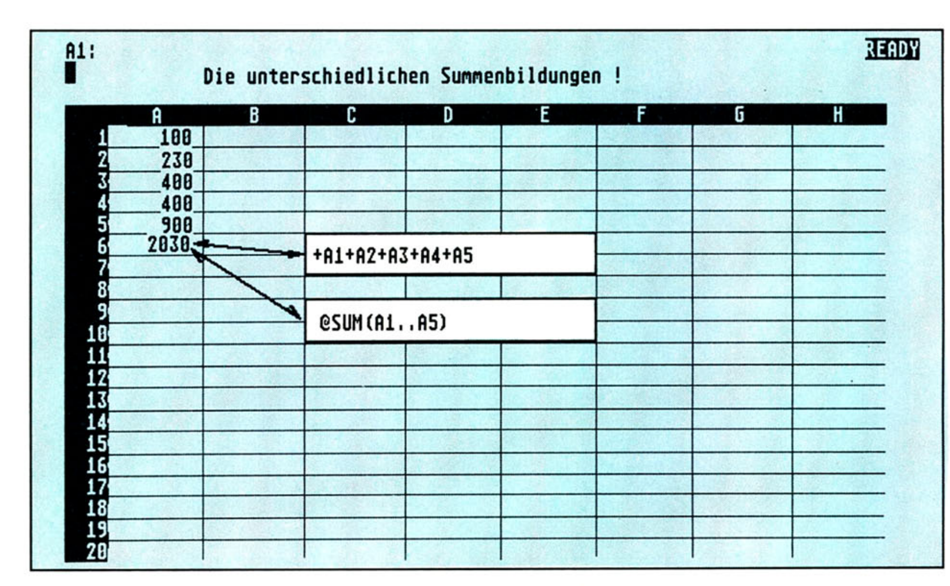


Bild 3: Summenbildung im VIP

Klappt doch schon ganz gut, oder? Üben Sie jetzt kräftig bis zum nächsten Teil unseres Programmierkurses, denn dann werden Ihnen die Formeln und Befehle nur so um die Ohren fliegen, als Vorgeschmack hier eine kleine Hausaufgabe. Geben Sie in eine beliebige Zelle nachfolgende Formeln ein:

1.) *+314+((7*1,987)^3)/765*
2.) *+2/0.185*(1258/8)*
3.) in Zelle A1 *15667* eingeben.
in Zelle A2 eingeben:
+A1(678/4)+(A1/2)*

HDS

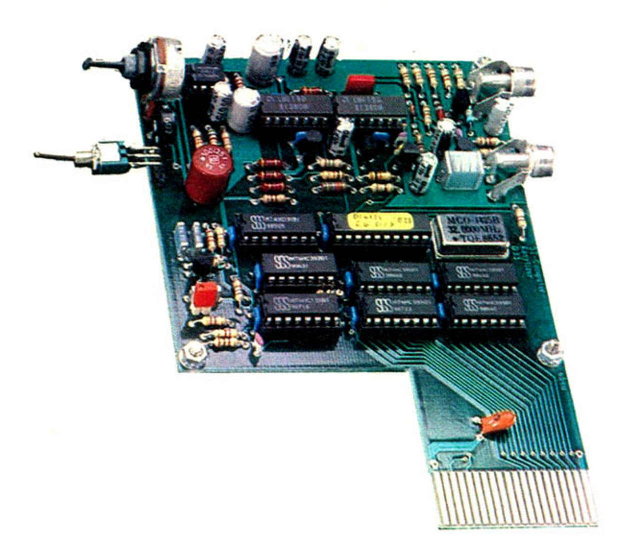
Zur Person des Autors:
Name: Heinz Dieter Schultz
Alter: 33
Ausbildung: Betriebswirt(VWA)
VIP-VIRUS: seit 1983 mit Lotus 1-2-3, über Symphony, dann zu Hause auf ATARI umgestiegen, Vip von Anfang an.
Bei Problemfragen: Heinz D. Schultz
Ingersheimer Str. 12
7140 Ludwigsburg
Bitte Versionsnummer VIP nicht vergessen.
(GEM oder TEXT, V1.1, V1.2 etc.)

TASTE:	*Bemerkung:*
@	Anweisung für Formel
S	der
U	Summen-
M	Befehl
(	Klammer öffnen, für den Bereich
Taste INSERT	END-Anweisung
CURSORTASTE (NORD)	Cursortaste und Sprung auf A1
.	Punkt (Anker) setzen
CURSORTASTE (SÜD)	Cursortaste
CURSORTASTE (SÜD)	Cursortaste
CURSORTASTE (SÜD)	Cursortaste
CURSORTASTE (SÜD)	Cursortaste
)	Klammer schließen, für den Bereich
RETURN	Mit Return die Eingabe abschließen.

Tabelle 1

Aus der Hardwareküche





Easytizer
– der Videodigitizer ohne Geheimnisse

Mit dem Easytizer können Sie beliebige Videosignale von einer Schwarzweiß- oder Farb-Kamera, Videorecorder oder auch direkt vom Fernsehgerät (mit Composite Video Ausgang) digitalisieren und somit auf dem Bildschirm Ihres ATARI ST sichtbar machen.

Der Easytizer wird am Modul-Port des ST angeschlossen. Die Auflösung beträgt 800x600 Bildpunkte, so daß in mittlerer Auflösung 640x200 Bildpunkte in vier Graustufen dargestellt werden können. In dieser Betriebsart werden **12,5 Bilder in der Sekunde** wiedergegeben! Im hochauflösenden Modus werden 640x400 Bildpunkte in Schwarzweiß wiedergegeben.

Besondere Merkmale des Easytizers:

- Software vollständig in Assembler
- Abspeichern der Bilder im DEGAS-Format
- Spiegeln eines Bildes in horizontaler und vertikaler Ebene
- Von STAD ansteuerbar
- Ein beliebiger Bildausschnitt kann in ein mit der Maus wählbares Format verkleinert oder vergrößert werden
- Animation mit beliebig vielen Bildern möglich, nur durch die Kapazität des Rechners begrenzt (beim Mega ST4 über 100 Bilder)
- Eingebauter Druckertreiber für die mittlere Auflösung für NEC P6/P7 und EPSON oder Kompatible
- Wahlweise automatische oder manuelle Helligkeitseinstellung
- Schnappschuß

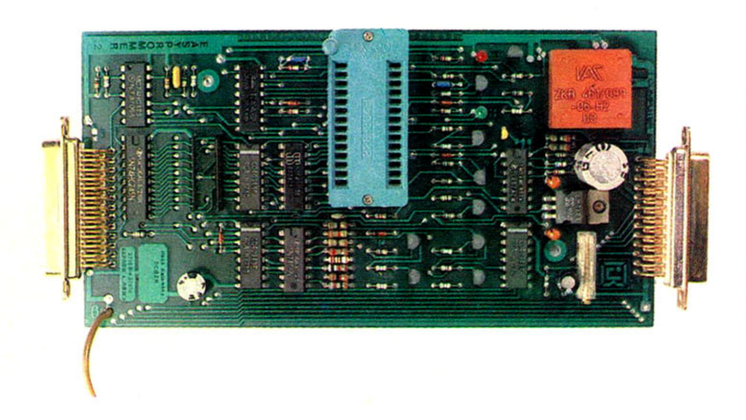
Lieferumfang:

1.) **Fertiggerät**
komplett aufgebaut und geprüft, inklusive Diskette mit der Easytizer Software und Bedienungsanleitung. Preis: DM 289,–

2.) **Teilesatz**
Für Bastler liefern wir einen Teilesatz bestehend aus:
- doppelseitiger, elektronisch geprüfter Platine mit Lötstopplack und Bestückungsaufdruck sowie vergoldeten Anschlußkontakten
- fertig programmiertes GAL 16V8
- Quarzoszillatormodul 32 MHz
- Diskette und Bedienungsanleitung — Preis: DM 129,–

Der Easyprommer V2.2

Ein universelles Programmiergerät für den ATARI ST, das nicht nur alle gängigen EPROMs der 27...-Serie inkl. 27011 (Megabit-Eprom) brennt, sondern auch die modernen EEPROMs lesen und programmieren kann. Die sehr komfortable Software, natürlich voll GEM unterstützt, erlaubt alle nur denkbaren Manipulationen. Mit dem integrierten HEX/ASCII-Monitor lassen sich Änderungen der Daten blitzschnell realisieren. Der auf der Platine befindliche Druckertreiber erlaubt das problemlose Drucken der Daten. Zwei Zusatzprogramme ermöglichen das Brennen **jeder** nicht kopiergeschützten Software direkt von Diskette einschließlich Accessories und Autostart-Programmen. Der Easyprommer wird an den Druckerport angeschlossen und samt Diskette und Handbuch geliefert.



Hiermit bestelle ich

☐ Easyprommer (Fertiggerät) für DM 349,–

☐ Easyprommer (Bausatz) für DM 298,–

☐ Easytizer (Fertiggerät) für DM 289,–

☐ Easytizer (Teilesatz) (w. o. beschrieben) für DM 129,–

☐ 128 KByte EPROM-Karte (fertig bestückt o. EPROMs) für DM 58,–

☐ Verlängerungskabel für Easyprommer für DM 39,90

Versandkosten: Inland DM 7,50 Ausland DM 10,–
Auslandsbestellungen **nur** gegen Vorauskasse

☐ Vorauskasse
☐ Nachnahme

Name: ____________ Vorname: ____________

Straße: ____________ Ort: ____________

Unterschrift: ____________

Bezugsquelle:



SOFTWARE DIE ES IN SICH HAT!

Nun auch in Deutschland!

Schwarz auf Weiß kann jetzt jeder auf seinem ATARI ST unter MS-DOS arbeiten. Doch nicht nur monochrom, nein, auch in Farbe ist nun der Zugriff auf die Welt der PC-Rechner möglich.

NEU EURO VERSION 3.64

Die Software-Emulation **PC ditto** öffnet allen ATARI ST Anwendern das Tor zum gewohnten professionellen Business Standard.

Mit dem **PC ditto** können Sie mühelos mit Lotus 1-2-3 oder Symphony Ihre Kalkulation erstellen, oder Ihre Daten mit DBase III plus verwalten.

Mit dem **PC ditto** haben Sie Zugang zu Turbo Pascal, zum GW BASIC Interpreter, und auch zu Borlands neuestem Kind Turbo Basic.

Mit dem PC ditto läuft auch die Software, die es für den ATARI ST noch gar nicht gibt.

Mit dem **PC ditto** laufen so viele Programme, daß wir sie hier gar nicht alle auflisten können.

Der **PC ditto** unterstützt die ATARI-Festplatte, den Druckerport, sogar den Laserdrucker und alle Schnittstellen, soweit es die Hardware zuläßt.

Machen Sie aus Ihrem ATARI ST den preiswertesten PC-Clone! Bestellen Sie den **PC ditto** zum Superpreis von nur DM 198,-.

Wir laden Sie ein! Steigen Sie mit Ihrem ATARI ST in die Welt des MS-DOS ein.

Hiermit bestelle ich ___ PC ditto für 198.-

Anruf genügt!
Tel.: 06196/481811, Mo-Fr 9-13 und 14-17 Uhr. Schriftliche Bestellung nur gegen Vorkasse oder Nachnahme (Versandkosten DM 7,50; bei Nachnahme zuzüglich DM 3,50 Nachnahmegebühr)

Name:____________________ Vorname____________________

Straße:____________________ Ort:____________________

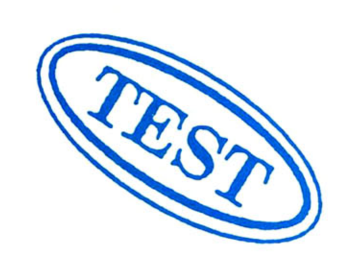
Unterschrift:____

MS-DOS und GW-BASIC sind Warenzeichen von Microsoft Corp. Lotus 1-2-3+ Symphony sind Warenzeichen von Lotus Dev. Corp. Turbo Pascal und Turbo BASIC sind Warenzeichen von Borland Corp. DBase III Plus ist ein Warenzeichen von Ashton-Tate Corp.

Nähere Informationen gegen ausreichend frankierten Rückumschlag bei:



MERLIN COMPUTER GMBH
INDUSTRIESTRAßE 26
6236 ESCHBORN
TEL.: 06196/481811



ZWISCHEN-LAGERUNG

Laut Murph'schem Gesetz treten sie immer dann auf, wenn man sie überhaupt nicht gebrauchen kann - die Zwangspausen, in denen man mal wieder auf den Drucker warten muß. Aber, wo eine Pause ist, ist auch ein Wille (oder so ähnlich), und so gibt es findige Helferlein, die der Warterei ein Ende bereiten sollen.

Obwohl fast alle Drucker, die heute auf dem Markt sind, eingebaute Zwischenspeicher von 2 bis 32 kRAM aufweisen, gibt es doch noch immer gewisse Wartezeiten beim Drucken, zumindest bei den Modellen mit bis zu 8 kRAM. Bei den größeren Zwischenspeichern läßt sich zwar eine Beschleunigung beobachten, doch wenn man mal eine falsche Datei zum Druck abgeschickt hat, kann entweder nur der unnötige Verschleiß an Papier und Farbband bedauert oder der Drucker durch Abschalten von seinen gespeicherten Daten befreit werden. Hier könnte durch einen von außen bedienbaren Resetknopf für den Speicher einiges anwenderfreundlicher gestaltet werden.

Speicher en gros...

Eine bessere Variante stellen da schon die Programme dar, die der heutigen Computergeneration mit ihren Riesenspeichern von 1 MByte und mehr ein paar kByte vom RAM abknapsen, um das Ganze zum softwaregesteuerten Druckerspooler umzufunktionieren. Das klappt aber nur so lange, wie die Programme normale Routinen zur Druckausgabe verwenden. Im anderen Fall werden nämlich diese Spooler einfach ignoriert und der gewünschte Effekt der Beschleunigung tritt nicht ein. Ein Paradebeispiel hierfür ist gerade das Textverarbeitungsprogramm SIGNUM!, das die Druckerdaten als reine Grafikdaten sendet und eine eigene Druckroutine benutzt.

Überlisten

Doch es gibt ja noch die dritte Variante, und die wollen wir hier einmal näher betrachten. Im konkreten Fall war es auch genau das Programm SIGNUM!, das die Versuche mit einem externen Druckerspooler oder Druckerpuffer interessant machte. Die externen Puffer sind in aller Regel kleine Kästen, die zwischen Computer und Drucker geschaltet werden, entweder als kleines Steckmodul (quasi als Verlängerung eines Centronics-Steckers) oder als eigenständiges Gerät mit separater Stromversorgung. Diese kosten je nach Speichergröße (64 kByte bis über 2 Mbyte RAM) ab etwa DM 400,-. Um so erfreulicher ist, daß es jetzt einen 64-kRAM-Spooler für knapp unter DM 200,- gibt, den wir hier einmal näher unter die Lupe nehmen wollen.

Aufgabenstellung

Wie gesagt, ausschlaggebend für den Test eines solchen Spoolers war die lange Wartezeit, die durch den Ausdruck von SIGNUM-Textdateien entsteht. Der Computer meldet sich ja wirklich erst nach dem letzten Pixelchen auf dem Papier zur Weiterarbeit zurück. Die Anlage, an der der Spooler getestet wurde, war ein normaler ST der alten Serie mit 1 MByte-RAM, daran angeschlossen ein NEC P6 mit automatischem Einzelblatteinzug. Als Testgrundlage dienten zwei Textdateien mit unterschiedlicher Größe, die als ASCII-Datei vom Desktop aus gedruckt und als Tetxe in SIGNUM! und Wordplus konvertiert wurden.

Outfit

Doch zuerst wollen wir uns das Gerät einmal näher anschauen. Der Multi-Buffer MB64 wird mit externem Steckernetzteil und 8-seitiger deutschsprachiger Gebrauchsanweisung in einem stabilen Pappkarton geliefert, deren äußere Werbesprüche die Lösung all unserer Probleme versprechen ("Drucken mit Buffer - Computer frei für gleichzeitige Benutzung" oder "speichert bis zu 64 k, ca. 32 Schreibmaschinenseiten" usw.). Nach dem Auspacken wäre mir das Gerätchen trotz seiner kleinen Abmessungen von 190 x 135 x 60 mm fast aus der Hand gefallen - das Ding hat das stolze Gewicht von 1,4 kg, wer rechnet denn mit sowas?... Zusammen mit den Gummifüßen ergibt das

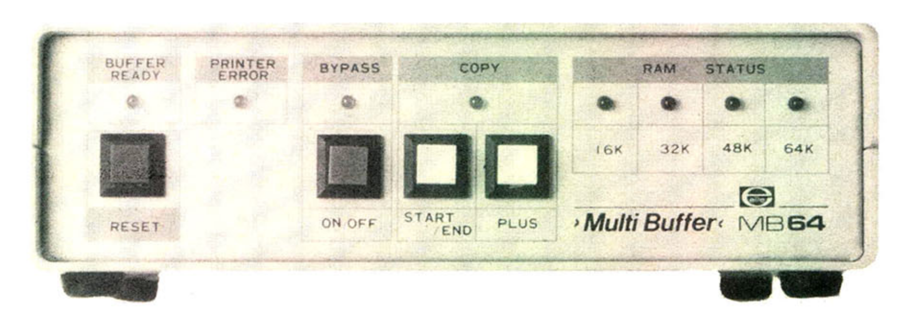


Bild 1: Die Bedienungselemente des Multi-Buffers

aber eine beeindruckende Standfestigkeit, die - schaut man sich die fingerdicken Centronicskabel an - gar nicht so verkehrt ist. Auf der Rückseite befinden sich neben den beiden Centronicsbuchsen der Ein/Ausschalter (nicht Netzschalter!) und die Buchse für die 9V-Spannungsversorgung. Hier wäre auch eine erste Kritik anzusetzen, die Klammer der Centronicsbuchse für den Printer läßt sich bei eingesteckter Spannungsversorgung nicht weit genug öffnen, um den Centronicsstecker an- oder abzukoppeln.

Tastatur

Auf der Vorderseite finden wir von links nach rechts den Reset-Knopf mit darüberliegender "Buffer-Ready"-Anzeige, eine Anzeige für "Printer Error", den Taster und die Anzeige für die "Bypass"-Funktion, eine LED für die "Copy"-Funktion mit den Tasten "Start/End" und "Plus" und vier LEDs, welche den "RAM-Status" in den Bereichen 16k, 32k, 48k und 64k anzeigen.

Gebrauchs-anweisung

Ich werde immer etwas mißtrauisch, wenn der erste Satz in einer Anleitung mit einer "Gratulation zum Kauf..." anfängt. Ist das jetzt ironisch gemeint, frei nach dem Motto "selbst daran schuld..." oder ehrlich, daß man es endlich geschafft hat, sein schlechtes Gewissen wegen der Geldausgabe zu überwinden? Naja, zumindest der zweite Satz verspricht wieder ruhigen Schlaf, der da heißt: "Jetzt können Sie endlich umfangreiche Dokumente ausdrucken und gleichzeitig an Ihrem Personal Computer weiterarbeiten". Die Anleitung selbst ist ausreichend und bietet auch dem Nur-Benutzer genügend Hilfestellung. Selbst eine Tabelle mit der Belegung der Ein- und Ausgangsbuchse fehlt nicht. Einer Adaption nicht ganz kompatibler Druckerports steht somit wohl nichts im Wege.

Innereien

Ein Blick in das Innere des Gehäuses verrät "gute Hausmannskost" und zeigt einen sauberen Aufbau. Dort versieht eine Z-80-CPU ihren Dienst (üblich, da sie den Refresh der dynamischen RAMs automatisch macht), neben dem 2764-EPROM mit der "Spooler-Software" gibt es noch einige TTL-ICs und die RAM-Bank mit acht 4164 (wider Erwarten liegen innen doch keine Bleiplatten, anscheinend ist das Gehäuse aus 2 mm Stahlblech...).

Jetzt wird's ernst

Schauen wir uns die Funktion des Spoolers mal im einzelnen an. Beim Einschalten leuchten die LEDs zur Kontrolle kurz auf und im RAM wird ein Selbsttest durchgeführt, die Anzeigen für den RAM-Status verlöschen dabei nacheinander (sie dienen später als Information für die Belegung des Speichers, bei "Überlauf" flackert die 64k-LED) und die Buffer-Ready-Anzeige leuchtet auf. Beim Druck auf die Reset-Taste wiederholt sich der ganze Vorgang und der Inhalt des Puffers wird gelöscht. Ist jetzt kein Drucker angeschlossen oder dieser angeschlossen, aber nicht eingeschaltet, oder aber eingeschaltet und Off-Line, so leuchtet zusätzlich die Printer-Error-LED. Bei dem NEC P6 mit Einzelblatteinzug wird das Papier erst kurz vor dem Ausdruck eingezogen. Drückt man beim Spooler allerdings die Bypass-Funktion, so zieht er das Papier sofort ein und wartet dann auf die auszudruckenden Daten.

Morse

An der Copyfunktion hätte der Erfinder der Telegrafie sicher seine Freude gehabt, denn er könnte den Puffer als Übungstaster benutzen. Das Aufrufen dieser Funktion funktioniert nämlich nach folgendem Schema:

1. Taste Start/End drücken (Copy-Anzeige blinkt)
2. Ausdruck am Computer starten
3. Für die gewünschte Anzahl der Kopien die Taste Plus entsprechend oft drücken
4. das Ganze mit nochmaligem Druck auf die Taste Start/End abschließen (Copy-Anzeige leuchtet)

Was sagten Sie gerade, wieviele Kopien brauchen Sie...?

Auf die Plätze...

Als erstes wurde die kleinere Datei (entspricht einer normalen Brief-Datei) einmal ohne eingeschleiften Spooler ausgedruckt und jeweils die Zeiten, bis der Computer wieder eingabebereit war, gestoppt. Anschließend wurde der Puffer angeschlossen und die Ausdrucke einmal über die Spooler-Funktion und ein zweites Mal über die Bypass-Funktion gedruckt. Die Ergebnisse finden Sie in der Tabelle 1. Die Zeitangaben für Wordplus stimmen nicht hundertprozentig, da man ja nach dem Druckbefehl eine andere Datei weiterbearbeiten kann (Druck im Hintergrund, es geht doch...). Die einzige Aussage über den Druckab-lauf ist das Häkchen im Menüeintrag - nur, wenn irgend ein Menüeintrag geöffnet wird, stoppt die Druckausgabe. Des-

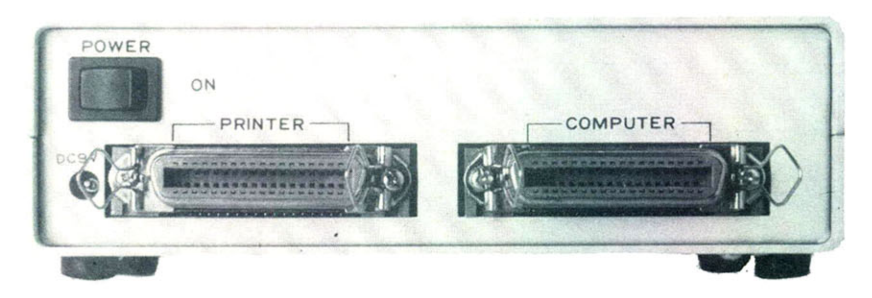


Bild 2: Der Multi-Buffer wird einfach eingeschleift

	Größe	ohne Spooler	Bypass	mit Spooler
ASCII	821	7,9 sek	2,8 sek	2,7 sek
Wordplus	942	19,3 sek	17,1 sek	9,8 sek
SIGNUM!	1980	44,5 sek	46,5 sek	25,8 sek

Tabelle 1: Zeitvergleich mit kleinen Dateien

wegen sind die Werte in Tabelle 1 etwas über den Daumen gepeilt.
Diese Werte muß ich kurz kommentieren. Der große Unterschied für die ASCII-Datei zwischen "ohne Spooler" und "Bypass" ergibt sich einfach daraus, daß beim Einschalten der Bypass-Funktion wie oben bereits erwähnt das Papier schon eingezogen wird. Ansonsten zeigen die Werte ja einen gewissen Zeitvorteil bei der Spooler-Funktion. Das beim SIGNUM!-Ausdruck allerdings die 16k-LED schon aufleuchtete, hätte mich mißtrauisch machen sollen...
Einen Spooler mit kleinen Dateien zu testen, ist ja nicht ganz im Sinne des Erfinders, deshalb wurde der Testlauf mit einem größeren Text wiederholt (ca. 5 Seiten ASCII-Text, s. Tabelle 2). Kaum zu glauben, aber SIGNUM! braucht ohne angeschlossenem Spooler weniger Zeit als mit! Außerdem war nach sage und schreibe 57 Sekunden der 64kRAM-Puffer voll...

Die restlichen Zeiten können sich sehen lassen, bei der ASCII-Datei meldet sich der Computer schon nach 12% der ursprünglich benötigten Zeit zurück, aber...

Fazit

Für eine Zeitersparnis beim Ausdruck von SIGNUM!-Texten ist dieser Druckerpuffer gänzlich ungeeignet - es heißt also weiterhin warten. Für die anderen Anwendungen ist er brauchbar, wäre da bei Wordplus nicht die Möglichkeit, eine andere Datei zu bearbeiten (oder dieselbe, die gerade gedruckt wird, nur unter einem anderen Namen), während der Drucker im Hintergrund läuft. Für ASCII-Dateien kann genausogut ein (bestimmt billigerer) Softwarespooler verwendet werden. Da wäre noch das Argument der Copy-Funktion, aber wie gesagt - schönen Gruß von Herrn Morse (wo ist denn der Softwarespooler mit Copyfunktion?).

HP

Bezugsquelle:

Elite Multibuffer MB64 -
Preis DM 199,-
erhältlich in allen Kaufhof-Filialen
(Außerdem notwendig:
Centronicskabel, gibt's ab
DM 30,- aufwärts)

	Größe	ohne Spooler	Bypass	mit Spooler
ASCII	18591	2'44 min	2'38 min	0'29 min
Wordplus	18931	5'52 min	5'46 min	2'35 min
SIGNUM!	31628	17'03 min	20'23 min	18'29 min

Tabelle 2: Zeitvergleich mit großen Dateien

ENDE

STEUERN SPAREN MIT

STEUER TAX'87

MIT DEN NEUEN 87ER VORSCHRIFTEN

DAS UNENTBEHRLICHE PROGRAMM ZUR RICHTIGEN BERECHNUNG DER LOHN- UND EINKOMMENSSTEUER FÜR ALLE STEUERZAHLER MIT ST-COMPUTERN IN DER BRD UND WEST-BERLIN

VERSION 2.7

MIT DEN NEUEN STEUERLICHEN ÄNDERUNGEN UND VORSCHRIFTEN FÜR 1987

- voll unter GEM eingebunden
- mausgesteuert, einfache Bedienung
- auf allen ATARI-ST-Rechnern lauffähig (bei 260 TOS im ROM)
- **Eingabe** an Steuerformulare **angepaßt**
- Auswertung auf Monitor oder Drucker wurde dem Steuerbescheid angepaßt
- schnelles durcharbeiten, da durch Pull-down-Menues nur die erforderlichen Bereiche bearbeitet werden müssen
- mit vielen Hilfen, so daß auch der Laie mit seinem ATARI schnell und mühelos seine Steuer berechnen kann
- ausführliches Handbuch, somit systematische Einführung in das Steuerrecht, **mit Steuertabellen und Tabellen für die Steuerklassenwahl bei Arbeitnehmer-Ehegatten**
- ständig werden aktuelle Steuer-Tips aufgrund der Einkommensteuerrechtssprechung eingebaut
- dem Handbuch sind Musterformulare beigefügt, um z. B. Werbungskosten aus unselbständiger Tätigkeit geltend zu machen
- Update-Service für die Folgejahre
- alle Eingaben und Auswertungen können abgespeichert und später wieder aufgerufen werden, um zwischenzeitliche Änderungen einzugben und Neuberechnungen durchzuführen
- die Version **2.7** ist geeignet für den „normalen Anwender", der für sich seine Steuer berechnen will
- S/W oder Farbmonitor

DM 98,–*

VERSION 3.7

MANDANTENFÄHIG

- Alle Merkmale wie Version **2.7**, jedoch zusätzlich mit einer Datenbank. Programm deshalb mandantenfähig
- pro doppelseitiger Disk können ca. 250 Mandanten abgespeichert werden, auf 20 MB Harddisk ca. 6.600!
- die Version **3.7** eignet sich besonders – aber nicht nur – für Steuerberater, Lohnsteuervereine, Buchführungshelfer, Versicherungsvertreter usw., die die Steuer auch für **andere** berechnen oder aber für solche Anwender, die mehrere Fallbeispiele für sich durchrechnen und abspeichern wollen
- darüber hinaus auch für Selbständige sehr interessant, die mehrmals im Jahr bzw. ständig einen Überblick über ihre Steuerbelastung haben wollen, um z. B. Investitionsentscheidungen zu treffen, also nach dem Motto: was muß ich noch tun, um die Steuerbelastung zu drükken **(was wäre wenn)**

DM 159,–*

UP-DATE SERVICE

STeuer Tax-Besitzer erhalten die neue Version **2.7** oder **3.7** gegen Rücksendung Ihrer registrierten Original-Diskette zum Preis von 35,– DM zuzügl. 5,– DM Versandkosten.
Lieferung erfolgt nur gegen Übersendung eines Schecks in Höhe von 40,– DM.

DM 35,–*

* Alle Preise sind unverbindlich empfohlene Verkaufspreise.

Bitte besuchen Sie uns in Halle 7 / Stand E 46



16. – 23. MÄRZ 1988

BESTELL-COUPON

an Heim-Verlag
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt

Bitte senden Sie mir **Lohn- und Einkommensteuer-Programm**

_______ St. STeuer-Tax 87 – Version **2.7** á 98,– DM   _______ St. STeuer-Tax 87 – Version **3.7** á 159,– DM
zzgl. DM 5,– Versandkosten (unabhängig von der bestellten Stückzahl)
☐ per Nachnahme   ☐ Verrechnungsscheck liegt bei

Name, Vorname ______________________________

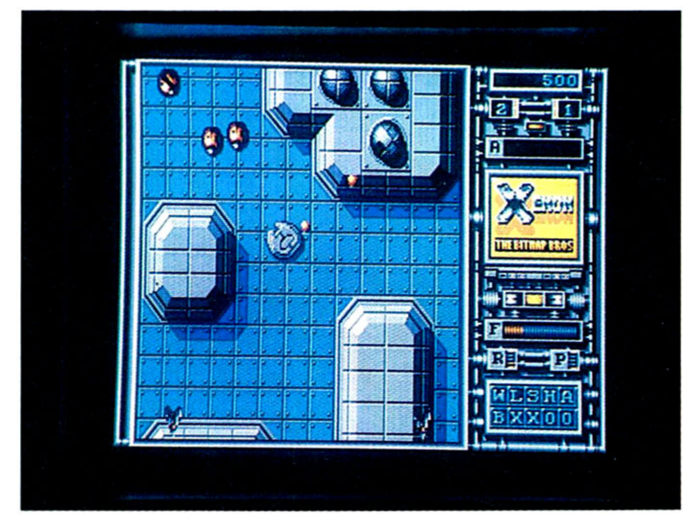
Straße, Hausnummer ______________________________

PLZ, Ort ______________________________
Benutzen Sie auch die im ST COMPUTER vorhandene Bestellkarte

Heim Verlag
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt
Telefon 0 61 51 - 5 60 57

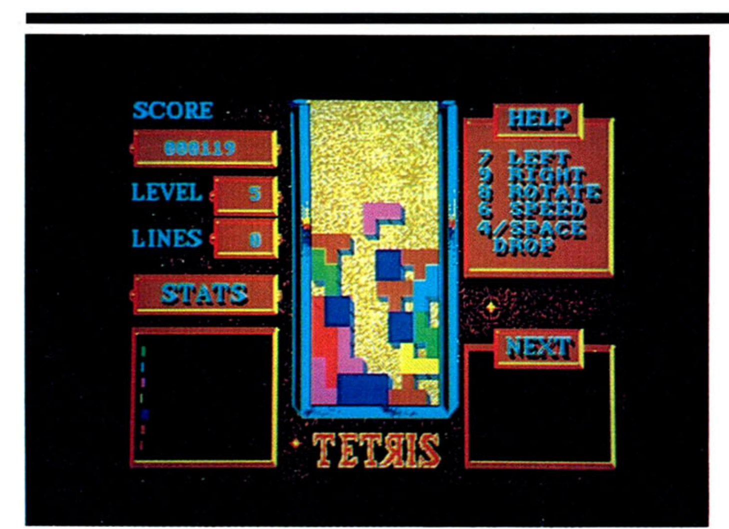
RELAX

XENON



Wieder einmal bleibt es an Ihnen hängen, Ihrem Captain aus einer brenzligen Situation zu helfen. Sie sind der einzige im Sektor und machen sich deshalb sofort auf, den - in Anbetracht der kurzen Zeit - langen Weg zurückzulegen. Sechszehn Zonen sind zu durchqueren, in denen starke Gengenwehr zu erwarten ist. Am Ende jedes Sektors wartet ein besonders gefährliches Kampfschiff - das 'Sentinel', das nur sehr schwer zu bekämpfen ist. Für diese Aufgabe steht Ihnen ein ganz besonderes Gefährt zur Verfügung, das Sie wahlweise als Bodenfahrzeug oder als schnelles Flugzeug verwenden können. Das Umschalten zwischen den beiden Betriebsarten erfolgt blitzschnell und gibt die Möglichkeit, sowohl die beweglichen Bodenfahrzeuge als auch die höhergelegenen Kanonentürme zu zerstören.
Einige der Kanonentürme hinterlassen nach ihrer Zerstörung Bonuspunkte, die man beim Überfliegen aufnehmen kann. Besonders wichtig sind dabei die Treibstoffpakete, die unbedingt in ausreichendem Maß mitgenommen werden müssen. Andere Punkte verschaffen Laserwaffen, Raketen oder andere schußstarke Waffen. Sehr wirkungsvoll sind dabei die 'Balls', die sich als Begleitgefährten an das Gefährt hängen und synchron feuern. Besonders im Kampf gegen den 'Sentinel' leisten diese Begleiter gute Dienste. Aber auch im Einsatz gegen die vielfältigen Angreifer, die so klangvolle Namen wie Ladybirds, Eyes, Spinners, Cells, Kites, Lip Guns haben, sind spezielle Waffen so gut wie unverzichtbar, wenn man den Hauch einer Chance haben will.
Obwohl es bereits eine Flut von 'Ballerspielen' gibt, ist XENON eine Bereicherung dieses Genres. Dies liegt daran, daß sich die Programmierer einige Gedanken darüber gemacht haben, wie man vom simplen 'Geballere' zum unterhaltsamen und spannenden Spielgeschehen kommt. Sehr gut gelungen ist deshalb die Steuerung, bei der man blitzschnell zwischen zwei Betriebsarten umschalten kann. Sehr effektvoll sind auch die Zusatzwaffen, die durch ihre knallige Aufmachung herrliche 'Baller'-Action garantieren.
Die Gestaltung des Spiels zeigt die Liebe zum Detail der Autoren und das wirkt sich sehr positiv auf die Spielfreude aus. Action- Fans sollten sich XENON deshalb nicht entgehen lassen.

Markus Nerding

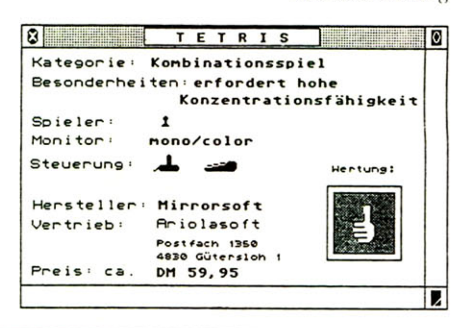


TETRIS

Angeblich kommt die Idee dieses Spiels aus den wilden Steppen des Urals, und wenn dem so wäre, dann sollte man mehr Spielideen von dort aufgreifen. Aufgabe des Spielers ist es, herabfallende Bausteine so zu drehen und anzuordnen, daß sie ineinander passen. Dabei gibt es acht verschiedene Steinformen, die durch die Farbgebung leicht zu unterscheiden sind. Fällt ein Stein herab, muß man blitzschnell feststellen, in welcher Lage er wohin am

besten paßt. Dabei kann der Stein so lange nach links und rechts bewegt und um jeweils 90 Grad gedreht werden, bis er den schon vorhandenen Steinhaufen erreicht. Wenn es gelingt, eine lückenlose waagrechte Reihe zu bilden, dann verschwindet diese und der Steinhaufen senkt sich um eine Reihe ab. Gelingt es nicht, lückenlose Reihen zu bilden, wächst der Haufen an, und wenn er das obere Feldende erreicht, ist das Spiel beendet.
Die Handlung von TETRIS ist im Prinzip sehr einfach, aber davon sollte man sich nicht täuschen lassen und zu dem (Kurz-)Schluß kommen, auch das Spiel sei einfach. Gute Kombinationsgabe und Reaktionsschnelligkeit sind hier gefragt, sonst kann man nur hilflos zusehen, wie der Steinhaufen 'in den Himmel wächst'. Wer sich etwas 'eingespielt' hat, wird zudem feststellen, daß sich mit diesem Spiel die Konzentrationsfähigkeit steigern läßt, denn nur damit kann auf die Dauer das nervenaufreibende Geschehen auf dem Spielfeld, bei dem man immer unter Zeitdruck reagieren muß, beherrscht werden. Somit empfehle ich dieses Spiel allen Freunden von schnellen Kombinationsspielen.

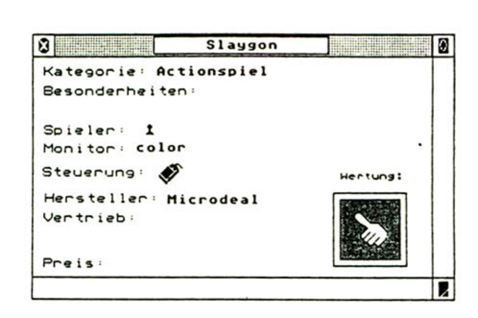
Markus Nerding



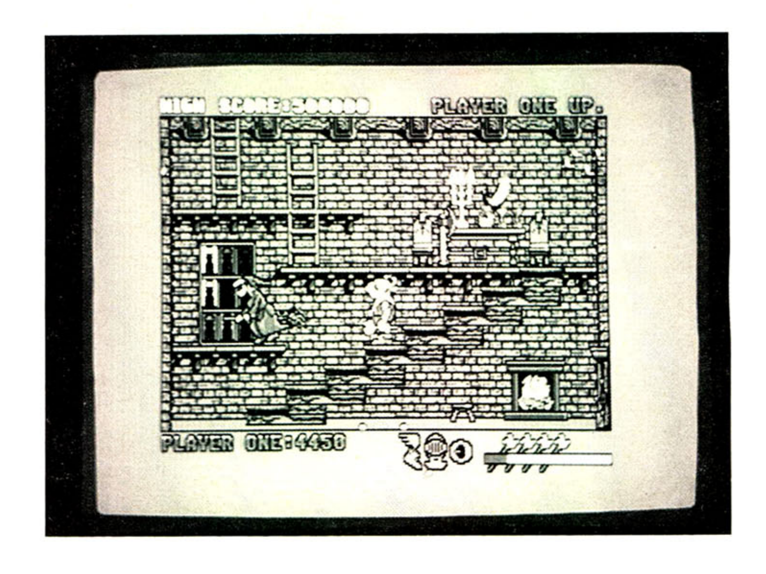
SLAYGON

Sie und Ihr Spezialfahrzeug werden gebraucht, um die Welt von einer unheimlichen Macht zu befreien. Diese will nämlich menschliche Arbeiter durch Roboter ersetzen; aber auch das ganze menschliche Leben könnte von dieser Macht ausgelöscht werden. Um Ihre Aufgabe zu erfüllen, gibt es nur eine Möglichkeit, Sie müssen das Kühlsystem des Kraftwerks zerstören, um das System so zur Explosion zu bringen. Das ist aber garnicht so einfach, Sie brauchen dazu nämlich fünf Codes, die Sie nur mit einem speziellen 'Interface' eingeben können. Sie dringen in das System ein, das aus einer Vielzahl von Gängen und Räumen besteht. Aber nicht jeden Raum kann man betreten, die Räume haben einen farblichen Schlitz an der Tür für eine Magnetkarte. Am Anfang des Spiels, wo man noch keinerlei Gegenstände besitzt, kann man nur in die Räume mit schwarzen Schlitzen. Außer schwarzen gibt es noch blaue, grüne, gelbe, rote und weiße Schlitze. Um in Räume mit blauen Schlitzen eindringen zu können, braucht man eine blaue Magnetkarte. Mit einer grünen Karte kommt man schon in Räume mit blau und grün gekennzeichneten Schlitzen. Mit einer weißen Karte kann man in jeden Raum eindringen. Räume können durch Roboter bewacht werden. Um sie zu umgehen oder unschädlich zu machen, haben Sie in Ihrem Fahrzeug einige hilfreiche Dinge zur Verfügung. Zum Beispiel den Cloaker, der Sie für die Roboter unsichtbar macht, oder den Lazer mit dem Sie den Roboter unschädlich machen können, um dann den Raum zu betreten. Auch versperren Energiefelder oder Minen Ihnen den Weg, dafür haben Sie dann einen Energieschild zur Verfügung. Aber Vorsicht, Ihre Spezialgeräte verbrauchen Energie und Ihr Energievorrat ist begrenzt. Besitzen Sie keine Energie mehr, ist die Mission fehlgeschlagen. Energie kann mit Energiebällen aufgefüllt werden.
Es gibt noch andere nützliche Dinge in den Räumen zu finden, zum Beispiel einen 'Force Field Neutralizer', der die Energiefelder unschädlich macht, oder Laserpistolen, die die Laserkraft erhöhen. Ihre Aufgabe ist ja, fünf Codes zu finden, die Sie dann im Terminal Room mit einen speziellen 'Interface' eingeben müssen. Ist dieses Ziel erreicht, heißt es, schnell den Ausgang zu finden, bevor das ganze System zur Explosion kommt.
Das Spiel besitzt eine übersichtliche Bildschirmaufteilung, in der Mitte die Richtungspfeile, die von der Maus angeklickt werden müssen. Im oberen Teil des Bildschirms befinden sich acht kleine Fächer, wo die gefundenen Gegenstände aufbewahrt werden können. Im unteren Teil befinden sich sechs kleine Bildschirme mit den Spezialgeräten des Fahrzeugs. Dann die beiden größeren Bildschirme, von denen der eine das System von oben zeigt und der andere die 'Geradeausrichtung' des Fahrzeugs zeigt.
Slaygon ist ein spannendes Spiel mit einer guten Idee, welches im 'Expert-Modus' schon mal einige Tage dauern kann.

Andreas Suchy



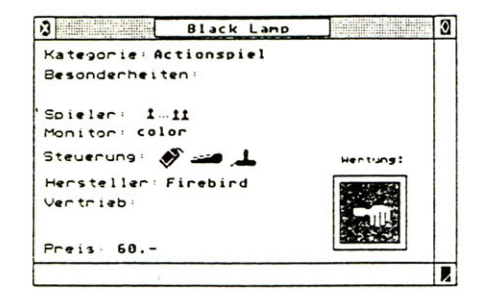
BLACK LAMB



Jack ist ein kleiner Hofnarr im Königreich von Allegorien. Seine Aufgabe ist es, Lampen zu finden und sie in das entsprechende Fach einer Truhe zu stellen. Jede Lampe hat eine eigene Farbe. An die schwarze Lampe ist am schwierigsten heranzukommen, sie wird nämlich von einem Drachen bewacht. Um sie zu bekommen, muß Jack erst den Drachen besiegen. Das macht er mit Hilfe seiner Zauberblitze. Aber Vorsicht vor den Feuerbällen, die der Drache ausatmet, denen muß Jack ausweichen oder mit seinen Blitzen abschießen.
Jack hat nicht nur Drachen als Gegner, so machen ihm auch Drachenfliegen, Wespen, Vampir-Fledermäuse, Werwölfe, Krähen, Bussarde, Kampfzwerge, Schleimhexen, Trolle, Schläger und Kobolde das Leben schwer. Sie können ihm dermaßen zusetzen, daß ihm der Energievorrat zur Neige geht. Aber zum Glück gibt es da die Speisen und Getränke, wo Jack seine Energie wieder auffrischen kann. Auch Juwelen, Musikinstrumente und Waffen lassen sich einsammeln, die zwar nicht die Energie auffrischen, aber den Punktestand erhöhen.
Die Steuerung erfolgt am besten über einen Joystick, aber auch mit der Maus oder der Tastatur läßt sich die Figur steuern. In Gebäuden und im Freien findet das Spielgeschehen statt. Die Gebäude haben mehrere Etagen welche man mit Leitern oder mit Springen über Kisten und Fässer erreichen kann.

Das Spiel ist in seiner Idee lustig und in der Grafik gut, aber die Fülle der angreifenden Gegner ist etwas störend und die Figur spricht nicht immer auf die gewollten Sprünge an.

Andreas Suchy


Black Lamp

Kategorie: Actionspiel
Besonderheiten:

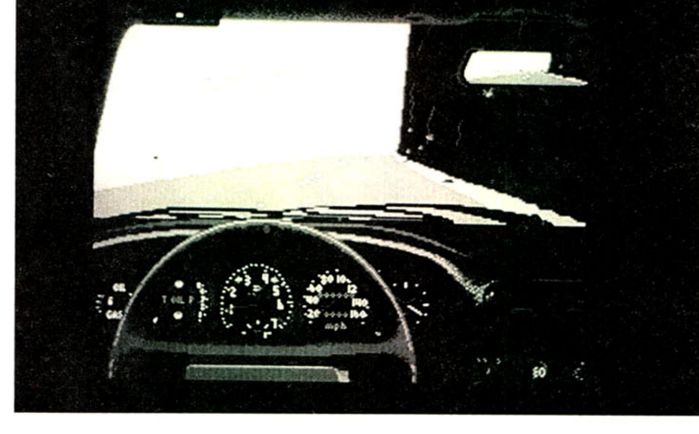
Spieler: 1...11
Monitor: color
Steuerung:
Hersteller: Firebird
Vertrieb:

Preis: 60.-

Wertung:

TESTDRIV E

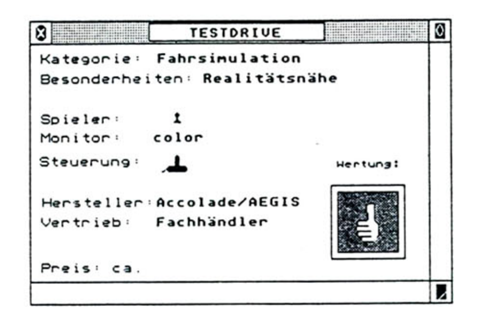
Mit Testdrive bringt Accolade den ersten Fahrsimulator für den ST auf den Markt. Ganz kurz zur Geschichte: Du hast Deine erste Million durch Softwareverkäufe gemacht und nun stehen Dir alle Türen offen, Dir Dein Auto, von dem Du immer geträumt hast, vor die Haustür zu stellen. Zur Wahl stehen mehrere Flitzer, denn das Hauptziel des Spieles besteht darin, möglichst schnell und unbeschadet von einer bis zur nächsten Tankstelle zu kommen. Die verschiedenen Fahrzeugtypen zählen zu den bekanntesten in der Welt: Porsche Turbo 911, Chevrolet Corvette, Lamborghini Countach, Lotus Turbo Esprit und Ferrari Testarossa. Zu jedem Fahrzeugtyp wird ein ausführliches Datenblatt über die Maschine gegeben, denn jedes Fahrzeug hat andere Eigenschaften. Sportliches Fahren (um nicht zu sagen, Geschwindigkeitsbegrenzung, was ist das?) ist angesagt. Das wird schon deutlich, nachdem der Traum wahr wurde und man sich für ein Fahrzeug entschieden hat. Ein smarter Fahrer läßt mal kurz die Sonnenbrille aus dem geöffneten Fenster blitzen und schenkt uns ein zuversichtliches Lächeln. Ab jetzt bist Du eins mit diesem Fahrer und hast die nötigen PS bzw. KW zu kontrollieren.



Up on the highway...

Vor sich die Armaturen und das Lenkrad, den Blick zuversichtlich aus der Windschutzscheibe auf die Straße, hat man seine Fähigkeiten und seine Erfahrung im Steuern mit dem Joystick zu beweisen. Fährt man zu langsam, bekommt man beim Zwischenstop prompt einen süffisant höhnischen Kommentar wie zum Beispiel frei übersetzt "Da fährt ja

meine Oma schneller!". Es gilt, bei Testdrive eine Menge Schwierigkeiten zu überwinden. Von Tankstelle zu Tankstelle begegnen einem immer neue Probleme, die einiges an Fahrgeschick erfordern. Kaum hat man sich halbwegs mit der Steuerung und der Gangschaltung seines Renners vertraut gemacht (am Anfang riskiert man leicht einen Motorschaden, wenn man nicht rechtzeitig schaltet), schon begegnen einem die ersten Trucks, PKWs, Ölflecken usw. Man muß Steinschlägen ausweichen und darauf achten, dabei nicht von der Straße abzukommen.


TESTDRIVE

Kategorie: Fahrsimulation
Besonderheiten: Realitätsnähe

Spieler: 1
Monitor: color
Steuerung:

Hersteller: Accolade/AEGIS
Vertrieb: Fachhändler

Preis: ca.

Wertung:

Ein Blick in den Rückspiegel

Unter anderem taucht auch immer wieder im Rückspiegel ein Wagen der Highway Police mit hübscher Partybeleuchtung auf, an dem man zuvor mit "Full Speed" vorbeigebraust ist. Da bleiben nur zwei Möglichkeiten, Gas geben und "ab durch die Mitte" oder brav überholen lassen, anhalten, den Strafzettel bezahlen und sich mit Sicherheit einen bissigen Kommentar an der nächsten Tankstelle abholen.

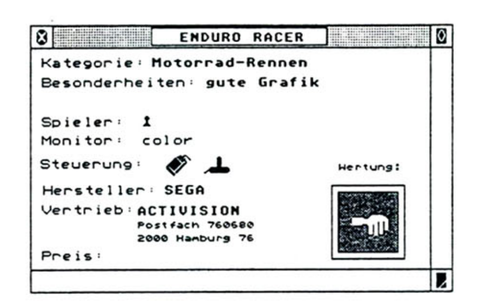
Im großen ganzen kann man Testdrive durchaus positiv beurteilen. Leider wird die Fahrerei auf Dauer etwas eintönig, da die Landschaft kaum Abwechslung bietet. Hier könnte bestimmt noch etwas getan werden, was das Spiel deutlich aufwerten würde. Auch in Bezug auf die Landschaftsgrafik wären noch Verbesserungen wünschenswert. Geliefert wird es auf einer Diskette und mit einer etwas mageren Anleitung. Doch der gewohnte Joystick-Artist wird auf Anhieb damit zurechtkommen.

HE

ENDURO RACER

Enduro Racer ist ein Motorradrennen nach dem üblichen Spielprinzip: Eine Straße, rechts und links davon Bäume oder andere Hindernisse und, immer in der Mitte: das Motorrad.
Es gibt fünf Levels zu überwinden. Für ein Level hat man sechzig Sekunden zur Verfügung. Von Level zu Level wird es etwas schwieriger, so gibt es zum Beispiel Sprunghügel zu überwinden, es liegen Steine auf der Straße, oder es kommt ein anderes Fahrzeug entgegen.




ENDURO RACER

Kategorie: Motorrad-Rennen
Besonderheiten: gute Grafik

Spieler: 1
Monitor: color
Steuerung:
Hersteller: SEGA
Vertrieb: ACTIVISION
Postfach 760680
2000 Hamburg 76
Preis:

Wertung:

Die Grafik ist durchschnittlich, allerdings ist die Steuerung alles andere als gut. Drückt man den Joystick nicht andauernd nach vorne, verliert das Motorrad an Geschwindigkeit und man kann den Level nicht schaffen, durch das andauernde Nachvornedrücken des Joysticks verliert das Motorrad aber an Wendigkeit, so daß man schneller gegen einen Baum prallt, als man denkt.
Die Motivation ist auch nicht besonders groß, nach mehrmaligen Spielen verliert man die Lust.
So kann man als Fazit sagen: Ein durchschnittliches Spiel mit keinen besonderen Vorkommnissen.

Andreas Suchy

Wollen Sie auch GELD an der Börse verdienen?

- *Verwaltung von bis zu **100** Aktien mit bis je **300** Kursen.*
- *Verwaltung von ausländischen Aktien mit bis zu **16** verschiedenen Währungen.* *(mit mathematisch genauer Einstandswährung.)*
- *Wertpapiere können alphabetisch sortiert werden.*
- *Mischen von bis zu **20** verschiedenen Depotdateien.*
- *5 verschiedene Kurzfristcharts und 2 verschiedene Langfristcharts.*
- *Auf Mausklick übereinanderlegen von verschiedenen Langfristcharts in weniger als **0,5** Sekunden (und natürlich wieder zurück).*
- *Charts können mit eigenen Kommentaren versehen werden. (beliebig viele!!!)*
- *Beliebig **gleitender Durchschnitt** (beliebig viele gleichzeitig).*
- *Sie können im Chart **zeichnen**. (Erkennen Sie Trendkanäle, Unterstützungen oder Widerstände.)*
- ***RSI-CHART** auf Mausklick. (RSI = Relative Stärke Index – damit arbeiten die Profis.)*



unverbindlich empfohlener Verkaufspreis

Erforderliche Hardware:
ATARI ST mit min. 1 MB Ram.
Monochrome Monitor
SF 354 oder kompatible.

- ***DEGAS** kompatibel (Sie können alle Charts auf Diskette abspeichern und mit Graphicprogrammen weiterbearbeiten.) Dies erleichtert Ihre Argumentationstechnik wesentlich.*
- ***DELUXE** unterstützt natürlich auch **Kapitalerhöhungen** und **Dividendenzahlungen.** (mit Operation Blanche!!!), und zeichnet diese natürlich im Langfristchart ein!*
- *Umfangreiches graphisches **Hilfsmenü** implementiert.*
- *2 verschiedene **Kurseingabe**möglichkeiten. (Für jeden Bedarf das Richtige!!!)*
- *Die wichtigsten Börsenusancen abrufbar.*
- *Alle Daten im Speicher = keine langwierigen Diskettenoperationen.*
- *Festplattenkompatibel.*
- *Programm wurde von **Aktienhändler** geschrieben.*
- ***Deutsches Handbuch. Up-date-Service.***

Heim Verlag
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt
Telefon 0 61 51 - 5 60 57

BESTELL-COUPON an Heim-Verlag, Heidelberger Landstraße 194, 6100 Darmstadt-Eberstadt

Ich bestelle: ☐ ***DEPOT DELUXE 398,– DM***
zuzügl. 5,– DM Versandkosten (Scheck oder per Nachnahme)
(unabhängig von bestellter Stückzahl)

Benutzen Sie auch die in ST COMPUTER vorhandene Bestellkarte.

Name: ____________________

Straße: ____________________

Ort: ____________________

Geheim-Tips

copySTar v2.2

- Kopiert **alle** Atari ST Disketten, **normale** und **kopiergeschützte** ohne zusätzliche teure Hardware.
- **Superschnelle Kopien** von "normalen" Disketten unter 30 Sek. Beim Kopieren wird das Format der Disketten (9,10,11 Sekt./SS-DS) automatisch erkannt.
- **Automatische Fehlerkorrektur** während des Kopiervorganges. Umfangreiche **Statusanzeige**.
- **Superschnelle Formatierung** z.B. SS 16 Sek. Erweiterung der Disk-Kapazität auf über 912KB ist mögl.
- Konvertiert in **Spezialformat für doppelte Geschwindigkeit** ohne zusätzliche Hardware.
- Prüfung von **Leer-Disketten** (**Qualitätstest**).
- **Testmöglichkeit der Drehzahl** des Disk-Laufwerkes.
- Super-**Update-Service**! Leichte GEM-Bedienung.

nur DM 169,–

☞PR's Disk-Speeder

- DAS Disk-Beschleunigungs-Programm für den ST.
- Durch ein intelligent **optimiertes Cache-Prinzip** erreichen Programme mit vielen Disk-Zugriffen **unglaubliche Geschwindigkeiten.** Und das ohne teure Hardware oder komplizierte Instalationen. **Frei konfigurierbar** (Device's/**Harddisk**/Speicherbelegung etc). Durch reine *Assembler-Programmierung* unglaublich schnell.
- Für den professionellen Anwender unentbehrlich.

nur DM 89,–

T.L.D.U. v1.3 The Last Disc Utility

- DIE komplette Disk-Utility mit wirklich allen Funktionen. Von vielen Fachzeitschriften für **SEHR GUT** befunden. Arbeitet natürlich auch mit Hard-und Ramdisk.
- Voll programmierbar durch 'C' ähnliche Makrosprache.
- Echte (Sub)-Directory und Fat-Funktionen mit Grafik.
- 10 Buffer, **Disassembler**, Folgemodus, Suchen, Vergleichen, Drucken, große Makro-Bibliothek u.s.w.
- Ein/Ausgabe in ASCII, EBCDIC, HEX oder DEZIMAL.

nur DM 149,–

Detective v1.0

- Erstellt editierbare **Assemblersource-Dateien** von jedem Atari-ST Programm.
- Verwaltet Text-, Data-, BSS-Bereiche, **Symbole**, Marker **und mehr**. Ein Muß für jeden Programmierer.

nur DM 149,–

Wichtig!! Alle Programme enthalten ein **ausführliches Handbuch** in Deutsch und keinen Kopierschutz.
Wo? Natürlich bei Ihrem Atari-Händler!
oder direkt bei

☆ STARSOFT ☆
Peinerstr. 81 D–3000 Hannover 81
☎ 05 11 / 8 37 99 77

TEAC
Made in Japan by Fanatics

Massenweise Massenspeicher von Profis für Profis und unglaublich Preiswert...

...und wir liefern auch weiterhin schnell und zuverlässig!!

Anschlußfertige Floppy-Stationen für ATARI-ST (Test in „ATARI SPECIAL" 1/88)

G3E-ST	3 1/2'	720 KB	**298,-**
G3S-ST	2★3 1/2'	2★720 KB	**598,-**
G5E-ST+	5 1/4'	720 KB / 360 KB.	**398,-**

(umschaltbar ATARI/IBM)

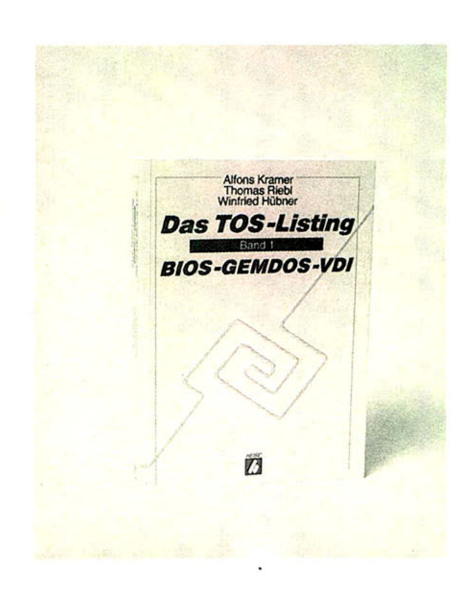
G35-ST+ 3 1/2' + 5 1/4'-Mixed-Station 2x720 KB, umschaltbar ATARI-IBM incl. Drive-Swap und Software **648,-**

Bestellannahme: Mo – Fr 8⁰⁰ – 18⁰⁰, Sa 8⁰⁰ – 12⁰⁰

Porto und Verpackung: Inland DM 7,50
Ausland DM 15,00

Versand Ausland nur Vorauskasse

***Copydata* GmbH**
8031 Biburg ★ Kirchstr. 3 ★ 0 81 41 - 67 97



Kramer, Riebl, Hübner
Das TOS-Listing Band 1
BIOS-GEMDOS-VDI
Heise Verlag, 368 Seiten
DM 68,-

Das Buch besteht hauptsächlich aus dokumentierten Listings von Teilen des TOS, sowie aus einer Einführung in die Arbeitsweise des GEMDOS. Es richtet sich damit an die fortgeschrittenen Programmierer und muß sich an deren Ansprüchen messen lassen.

Zu den Listings ein paar allgemeine Bemerkungen: Positiv zu bewerten ist, daß sie sich alle auf das Blitter-TOS beziehen, was sie gerade für MEGA-ST-Besitzer interessant macht. Aber auch für Benutzer des alten TOS sind sie sicher brauchbar, da damit zu rechnen ist, daß sich das neue TOS auf lange Sicht durchsetzen wird.

Die Kommentare sind zahlreich bemessen und die Listings strukturiert, d.h. der Beginn von Funktionen ist hervorgehoben, oft mit Angaben über Register oder Parameter. Ärgerlich sind nur die vielen Druckfehler, angefangen von der Rechtschreibung bis zu fehlerhaften Adressen im Disassembling (!) und vertauschten Zeichen (wie "$" und "#"), was zu großen Mißverständnissen führen kann. In den 'C'-Listings finden sich manchmal noch direkte Adressen an Stelle von Variablennamen. Dies alles deutet darauf hin, daß die Listings nur unvollständig ediert wurden.

Bei der Bezeichnung von Funktionen und Variablen wurden schon veröffentliche Namensgebungen übernommen, was zu loben ist. Bei den Systemvariablen (dokumentierte und undokumentierte), von denen eine - allerdings unvollständige - Liste vorhanden ist, wurden oft fantasielose Namen wie "L8846" vergeben.

Ungünstig ist auch, daß oft Adressen durch symbolische Bezeichnungen ersetzt wurden, aber der Assemblercode selbst fehlt. So sind Verzweigungen öfters nicht nachvollziehbar. Es wäre sicher besser gewesen, direkte Adressen und Symbole anzugeben.

Das BIOS-Listing ist der umfangreichste Teil des Buches. Deutliche Ähnlichkeiten mit einem anderen bekannten BIOS-Listing sind nicht zu verleugnen, einige der dort vorhandenen falschen Kommentare sind korrigiert, andere hingegen wortwörtlich identisch!

Praktisch ist, daß die in 'C' programmierten Routinen des BIOS auch als 'C'-Listing vorhanden sind, was das Verständnis erleichtert.

Garniert ist die BIOS-Dokumentation mit einigen nützlichen Extras wie Listings des RAM-TOS-Boot-Sektors oder der Desktop-Formatierroutine. Auch eine Anleitung zur Programmierung resetresidenter Programme wurde ergänzt.

Die Einführung in GEMDOS umfaßt wesentliche, aber knappe Informationen über dessen innere Struktur. Sie ist wohl nur als Ergänzung zum eigentlichen Listing zu verstehen, dabei aber sicher hilfreich, da GEMDOS im Detail nicht so einfach zu durchschauen ist.

Bei der vorgenommenen Bewertung des GEMDOS kann ich Herrn Kramer allerdings nicht folgen. So werden GEMDOS nur wenige konzeptionelle Schwächen attestiert während den vielen Fehlern nur eine (!) Seite gewidmet wird. Über die dort behauptete Quasi-Fehlerfreiheit kann man nur staunen, wenn man seine eigenen Programmiererfahrungen mit GEMDOS betrachtet.

Das Listing ist, wie GEMDOS selbst, in 'C' verfaßt, d.h. es wurde per Hand "rekompiliert". Es ist ausreichend kommentiert und hat sogar eine Referenztabelle für die einzelnen Funktionen und Variablen. Es wurden bisher keine groben Fehler festgestellt, nur wurden bei der Kommentierung oft selbst sehr offensichtliche Programmierfehler übersehen. Es dürfte einem große Einblicke in die Funktionsweise von GEMDOS geben.

Das sogenannte "VDI-Listing" verdient diesen Titel nicht. Auf 40 Seiten findet sich nämlich nur das Listing (Assembler) einer guten Hälfte der "Line A-Routinen" und ganz weniger VDI-Funktionen. Darin sind zwar die neuen Blitter-Routinen enthalten, was sicher für die Blitter-Programmierung nützlich sein dürfte; das eigentliche VDI jedoch fehlt praktisch!

Wie in jedem in letzter Zeit neu erschienenen Buch gibt es eine Dokumentation zum Blitter, wie gewohnt in Anlehnung an die Atari-Blitter-Dokumentation.

Ferner enthält das Buch auch ein Listing des Atari-Harddisk-Treibers, der zwar nicht zum TOS gehört, aber dessen Studium doch recht interessant und lehrreich sein kann. Obwohl am kürzesten, gefällt mir dieses Listing mit Abstand am besten. Es ist englisch, dafür aber sehr schön kommentiert und weist mit Witz auf die auch hier unvermeidbaren Programmierfehler hin. Es beinhaltet ebenfalls eine leicht verständliche Erläuterung der einzelnen Komponenten der Harddisk und deren Zusammenwirken.

Zusammenfassend ist zu sagen, daß sich das Buch wohl für jemanden lohnen könnte, der noch über kein BIOS-Listing verfügt oder besonderes Interesse am GEMDOS, dem Harddisk-Treiber oder der Blitter-Programmierung hat, vorausgesetzt, er ist in der Lage, mit den offensichtlichen Fehlern zurechtzukommen.

Aufgrund der zahlreichen Ungereimtheiten sollte man jedoch nicht sein altes BIOS-Listing wegschmeißen und sich dieses kaufen, besonders nicht in Anbetracht des nicht gerade geringen Preises.

Andererseits kann man froh sein, daß es überhaupt veröffentlichte Listings vom GEMDOS und Harddisk-Treiber gibt, und das sogar für das Blitter-TOS.

Alex Esser



Batz, Huschke
Sortierverfahren in Pascal
Franzis' Verlag, München
80 Seiten, 1. Auflage 1988
DM 28,-

Bei den meisten Programmieraufgaben benötigt man sortierte Daten. Dies stellt bei kleinen Datenmengen kein großes Problem dar, denn jeder wird ohne viel Nachdenken den Bubble-Sort-Algorithmus einsetzen können. Werden die Datenmengen jedoch größer, benötigt dieses Verfahren sehr viel Zeit. Abhilfe schaffen dann die effizienter arbeitenden Algorithmen wie Shell/Metzner-Sort, Heap-Sort oder der bekannte Quicksort. Das Buch beginnt mit einer kleinen Einführung über Zeiger und Rekursion in Pascal. Danach werden die einfacheren Sortieralgorithmen vorgestellt, von denen jedoch nur der Shell-Sort sinnvoll erscheint. Interessanter sind natürlich die Versionen des Heap- und Quicksort, von denen das nächste Kapitel handelt. Leider ist jedoch die Erklärung des Quicksorts sehr kurz ausgefallen, obwohl dieses Verfahren erwiesenermaßen das schnellste ist. Apropos Geschwindigkeit, vergeblich sucht man einen Zeitvergleich der vorgestellten Programme, und auch die Berechnungsformeln für die Anzahl der Durchläufe werden nicht angegeben. Auffallend ist auch, daß die einfachen Verfahren in fast allen Variationen aufgeführt werden, bei den fortgeschrittenen jedoch optimierte wie z.B. Supershell oder das Shell-Metzner-Verfahren unerwähnt bleiben. Ähnliche Schwächen zeigt auch das abschließende Kapitel über Suchbäume auf. Die Dokumentation ist ziemlich unglücklich gewählt, außerdem fehlen hier, wie im gesamten Buch, Struktogramme der vorgestellten Routinen. Auch auf ein Beispielprogramm hat man verzichtet.

Das Buch kann nur für Einsteiger empfohlen werden, die in erster Linie einen der vorgestellten Sortier-Algorithmen in Pascal benutzen wollen, aber dafür ist das Buch zu teuer. Für Interessierte fehlen Struktogramme, Geschwindigkeitsbetrachtungen und nähere Erläuterungen zu den Algorithmen.

Markus Nerding

Leser -> ST

Betr.: ST Computer 3/88 "Vom Quelltext zum Programm"

Leider hat sich in das Listing des neuen Linkers für das GST-Format ein kleiner Fehler eingeschlichen. In der drittletzten Zeile muß statt

```
if(listing_flag) fclose(fp);
```

richtigerweise folgendes stehen:

```
if(listing_flag) fclose(list_file);
```

Ingo Eichenseher, Stadtbergen

Betr.: Zeitmanager PD-Diskette 113

Das Programm gefällt mir wirklich gut, doch macht es mir einige Sorgen, da es schlicht und einfach den 29.Februar 1988 ignoriert. Es wäre nicht weiter schlimm, denn alles neu macht der März (?!), doch liegen alle Folgetage des Jahres um einen Tag falsch. Was kann ich tun?

Bernd Baumann / Wuppertal

Red.: Bei solch komplexen Programmen kommt es gerne vor, daß etwas nicht korrekt funktioniert. Da auf der Diskette der Source-Code des Zeitmanagers enthalten ist, können Sie die Probleme durch folgende Änderungen aus der Welt schaffen. Ersetzen Sie folgende Programmteile durch die angegebenen Anderungen. Die Stellen im Programm (hier wären Zeilennummer nicht schlecht), finden Sie mit Hilfe der Such-Funktion.

1.) Erkennung auf Schaltjahr

```
If Jahr% Mod 400=0
 Schaltjahr%=-1
Endif
If Jahr% Mod 100=0
 Schaltjahr%=0
Endif
```

durch:

```
If Jahr% Mod 1=0
 Schaltjahr%=0
Endif
If Jahr% Mod 4=0
 Schaltjahr%=-1
Endif
```

2.) Kalenderanzeige bis 29.02.

```
For I%=1 To M1%(M%)
```

durch:

```
For I%=1 To M1%(M%)-
     (M%=2 And Schaltjahr%)
```

(Diese Zeile ist zweimal zu ersetzen)

3.) Berechnung der Tage

```
If Mo%=2 And Ta%>28
  Ta%=28
Endif
```

durch:

```
If Mo%=2 And Ta%>28-Schaltjahr%
 Ta%=28-Schaltjahr%
Endif
```

Wir hoffen, daß damit alle Fälle abgedeckt sind. Das Jahr 1988 ist nun korrekt und der 29. Februar wurde zum Leben erweckt.

Betr.: ST Computer 3/88 "Schreiben mit dem WRITER-ST"

Zum einen ist es selbstverständlich möglich, ohne große Umwege ein neues Dokument zu erzeugen. Sie können sofort nach dem Aufruf von WRITER-ST mit dem Erstellen des Textes anfangen. Erst wenn sie den Text fertig geschrieben haben, können sie diesen unter Angabe eines neuen Dateinamens abspeichern.
Zweitens wird in dem Testbericht der Eindruck erweckt, daß es unmöglich sei, einen Textblock in die Mitte einer Zeile einzufügen. Dies ist selbstverständlich möglich, dabei werden aber die Wörter, die nicht mehr in die zu bearbeitende Zeile passen, in eine neue Zeile geschoben. Sollen gleich mehrere Wörter eingefügt werden, so kann man sich durch "CTRL/Return" ein wenig Platz schaffen.
Der Mangel, daß dabei für jedes herausgeschobene Wort eine neue Zeile eingefügt wird, ist in der Version 1.2 behoben.

Martin Schmitt-Degenhardt, SSD-Software

Betr.: ST Computer 3/88 "JuriSTische Anwendungsprogramme"

Zu Ihrer März-Ausgabe der ST-Computer möchte ich Ihnen recht herzlich gratulieren.
Nach dem schleichenden Niveauverlust der beiden vorherigen Ausgaben des Jahres 1988 dachte ich, die ST-Computer würde langsam aber sicher zu einem Werbeblättchen ohne besondere Ansprüche verkommen. Insbesondere das Layout erinnerte mehr an das Schinden von Seiten als an die Vermittlung sachlicher Informationen. Teilweise waren die Anzeigen so aufgemacht, daß sie wie Teile von Artikeln erschienen, teilweise sah ein Artikel eher wie eine Werbeanzeige aus.
Gott sei Dank habne Sie nun Ihre Layoutversuche eingestellt und zu Ihrem ursprünglichen Bild zurückgefunden. Sachliche Informationen werden wieder in einem entsprechenden Rahmen vermittelt.
Insbesondere weckt Ihr Artikel über juristische Anwendungen mein Interesse. Wir sind hier mit mehreren Jurastudenten zusammen, die einen ATARI ST zu Studienzwecken einsetzen. Nachdem wir bereits Hausarbeiten etc. erstellt haben und dabei hinreichende gute als auch schlechte Erfahrungen mit diversen Textverarbeitungsprogrammen sammelten, blieb uns als juristische Anwendung hauptsächlich noch, den ATARI ST als großen, blitzschnellen Karteikasten zu nutzen.
Die Möglichkeit, den ST dann auch später weiter einzusetzen, ist deshalb für uns alle nicht nur aus finanziellen Gründen besonders reizvoll. Nachdem sich in der letzten Zeit einiges auf dem Sektor der Netzwerktechnik zu tun scheint, stellt Ihr Artikel nun wohl auch einen Ansatz dar, sich mit anwaltsgerechter Software öffentlich auseinanderzusetzen und nicht auf reine Mundpropaganda angewiesen zu sein oder sich alles selbst zusammenzusuchen.
Ich hoffe daher ebenso wie meine Bekannten, daß Sie den Artikel wie angekündigt zu einer kleinen Serie ausbauen und verschiedene Programme für den juristischen Bereich vorstellen.
Als Anregung könnte man Ihnen vielleicht raten, in Zukunft neben einer Serie für Juristen auch solche für andere nicht naturwissenschaftliche

Bereiche herauszubringen. In diesen Bereichen ist die Angst vor der Technik relativ groß und gerade ein preiswertes System wie der ATARI ST für einen Einstieg geeignet. Oft ist es den Betroffenen jedoch nicht möglich, sinnvolle Anwendungen für Computer zu erkennen, oder Ihnen wird eine völlig falsche Beratung zuteil. Händlerauskünfte wie "Die Mindestinvestition für eine reine Textverarbeitung liegt bei ca. 20.000,- DM, alles andere ist einreines Telespiel und für somit nicht geeignet," sind üblich und schrecken viele potentielle Anwender grundsätzlich ab. Gerade im Bereich der voll- oder halbprofessionellen nicht naturwissenschaftlich-mathematischen Anwendung des ATARI ST fehlt es noch völlig an einschlägiger Literatur und aktuellen Nachrichten, so daß gerade hier vielleicht ein potentieller Käuferkreis wartet, den Sie mit dem Niveau Ihrer März-Ausgabe sicher erreichen können.

Thomas Giese, Bonn

Red.: Erst einmal vielen Dank für Ihr Lob, doch bleiben wir auf dem Teppich; bei uns hat sich viel getan. So haben die massiven Umstellungen in der Layoutgestaltung und der Produktion doch nicht so reibungslos geklappt, wie wir es uns vorgenommen hatten. Dies hat sich dann recht deutlich auf die Januar- und die Februar-Ausgabe und somit leider auch auf unsere Leser ausgewirkt. Mit dieser April-Ausgabe dürften wir aber wieder alles unter Kontrolle haben und hoffen, daß die ST-Computer auch weiterhin Anklang findet.
Zum zweiten Teil ist zu sagen, daß wir auch in Zukunft über Anwendungen des ST in der Praxis berichten wollen. Allerdings wird das wohl eher in Form von ein- oder zweiteiligen Artikeln geschehen. Denkbar wären zum Beispiel Berichte über den Einsatz des ST in Arztpraxen, in Meßlabors oder in der Schule. Die Berichte werden nicht in jeder Ausgabe erscheinen, sondern in unregelmäßigen Abständen. Wir sind der Meinung, daß wir so unseren Lesern auch die Möglichkeiten der sinnvollen Nutzung von Computern zeigen können, und somit ist auch jeder Leser aufgerufen uns seine Erfahrungen mitzuteilen, damit alle davon profitieren können.

Leser -> Leser

Betr.: Druckertreiber für Brother CE-550

Ich wäre sehr dankbar, wenn mir jemand bei meiner Suche nach einem Druckertreiber für meine Brother CE-550 und den ATARI ST 1040 weiterhelfen könnte. Ich möchte speziell 1st_Word einsetzen. Alle Treiber, die ich besitze, laufen nicht. Ich hoffe, daß mir ein Leser helfen kann.

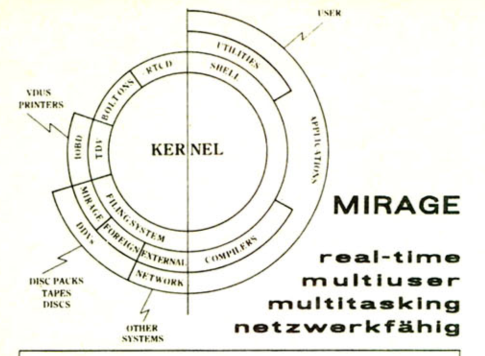
Wulf D. Richter, Postfach 280426, 1000 Berlin 28

Betr.: Leserbrief ST-Computer 3/88 "Probleme mit 1st_Word Plus und Etikettendruck"

Im oben genannten Leserbrief erwähnt Hanno Sekuterski aus Arnsberg seine Probleme beim Etikettendruck mit 1st_Word Plus.
Die Lösung, die angeboten wurde, ist zwar auch nicht schlecht, aber es geht meiner Meinung nach noch einfacher.
Man stellt im Menü Seitenformat die Papierlänge auf die entsprechende Etikettenlänge ein und schreibt dann ganz einfach seinen gewünschten Text für den Aufkleber. Man muß nur darauf achten, am Textende keine neue Seite anzufangen, also nicht in den linken senkrechten Balken zu klicken.

Uwe Ehrhardt, Freiburg

ENDE



MIRAGE

real-time
multiuser
multitasking
netzwerkfähig

Die Alternative zu UNIX. Seit Jahren hat **MIRAGE** sich auf dem Markt der Workstations fest etabliert. Jetzt gibt es die ST-Version. Extrem schnell. Extrem vielseitig. Einige Features:

- Lesen & Schreiben von TOS-Disketten
- TOS-und **MIRAGE**-Partitions auf **einer** Festplatte
- TOS-Calls von **MIRAGE** aus möglich (**kein** GEM)
- Line-A Grafik zugänglich
- virtuelle Bildschirme (umschaltbar per Tastatur)
- bis zu 128 dynamische Tasks pro Job
- Kommandosprache mit Variablen, Nesting etc.
- geräteunabhängiges Treiberkonzept
- schneller, komfortabler Editor
- Supervolumes: Eine Datei über mehrere Drives
- echtes Netzwerk ohne zusätzliche Hardware
- Sprachen: Basic, Pascal, Lisp, Fortran, C, APL

Der **Assembler**:

- Macros (mit lokalen Labels und Konstanten!)
- bedingte Assemblierung
- Unterstützung von 68020/68030/68881/68882
- Fließkomma-Konstanten
- Linker, Debugger, Profiler, Library-Funktionen

MIRAGE: Die ideale Lösung für Software-Entwickler, die eine preiswerte Workstation-Umgebung brauchen.
MIRAGE: Für Unis, die z.B. ihre Fortran-Anwendungen aus dem Rechenzentrum lösen wollen (Swifte-Fortran -77 ist als **Error-Free** zertifiziert, die **NAG**-Bibliotheken sind erhältlich - auch auf Disketten!).
MIRAGE: Für Meßdatenerfassung & Steuerzwecke.
MIRAGE: Für die Portierung von PC-Software auf den ST (z.B. mit Lattice C); für Netzwerke; ...

MIRAGE-ST (ROM-Pack, Disks, Manuals)	**348.-**
MicroGrid (Netztreiber für RS232 u. MIDI)	**98.-**

- Bitte fordern Sie Infos und Preislisten an -

gdat Stapelbrede 39 4800 Bielefeld 1 **0521/875 888**

We don't need no education



We don't need no thought control

APL/68000 - Das ganz besonders schnelle APL für 68000er Rechner. **Von MicroAPL.** Die Sprache der Profis für ATARI, MAC und AMIGA - oder darf es eine Workstation sein? Bei **gdat** zum supergünstigen Preis: Incl. Support, Update-Service und Toolsdiskette (nur ST). **Wir liefern immer die neueste Version.** Literatur- und Referenzhinweise sowie ausführliche Infos gratis.

APL/68000-ST+ — **298.-**
Die PLUS-Version wird exklusiv von **gdat** vertrieben und enthält wertvolle Erweiterungen und Tools, darunter sogar eine kleine Datenbank!

APL/68000 Language Manual — **49.-**
Der Kaufpreis wird bei nachfolgendem Kauf des APL voll angerechnet.

APLPRINT — **198.-**
druckt Text und Grafik unter APL.

LineAl — **164.-**
Schnelle Line-A Grafik.

APL-ASS — **164.-**
Für Assembler-Routinen in APL.

APL-EDIT — **248.-**
Variablen-Editor, Icons, Fonts ...

GD_GRAPH — **48.-**
VDI-Grafiktools.

APL ist eine Hochsprache der vierten Generation, die seit Jahren ihre Effizienz erwiesen hat. Und: APL ist eine Sprache zum Anfassen - Leicht zu lernen, intuitiv, fehlertolerant. Mit APL werden Probleme gelöst, nicht geschaffen. Zudem können wir uns als zuverlässigen Partner empfehlen: **gdat** arbeitet seit 7 Jahren mit APL/68000. Wir wissen, wovon wir reden. Fordern Sie das Info an.

gdat Stapelbrede 39 4800 Bielefeld 1 **0521/875 888**

Call Soft
COMPUTERSYSTEME
Verkauf und Versand, Beratung, Branchenlösungen

1040 STF, komplett mit Monitor SM 124	**1498,-**
ST 520 STM, 512 Kb RAM, Maus, TOS-ROM's	579,-
MEGA ST-2 / MEGA ST-4	SUPERPREISE auf Anfrage
LAUFWERKE	
ST-Doppelfloppy 2x726 Kb, eingeb. Netzteil	598,-
ST-Einzelfloppy wie vor, jedoch 1x726 Kb	329,-
40 MB Festplatte HD 40, anschlußf. Bootfähig	2198,-
20 MB Festplatte SH 205, anschlußfertig	**1198,-**
MONITOR	
NEC MULTISCAN **GS** NEU, alle ST Aufl.	598,-
SM 124 Monochrom ... 469,- / SC 1224 Color	869,-
DRUCKER	
NEC-P2200, NEU, 24 Nadeln, Bi. Traktor, 55z LQ	995,-
NEC-P6, 24 Nadeln, 216z Draft / 72z LQ	1258,-
NEC-P7, wie P6, jedoch dIN A3	1658,-
Farbdrucker NEC-CP6 .... 1558,- / NEC-CP 7	1958,-
EPSON LQ-850, 24 Nadeln, 226z Draft/88z LQ	1698,-
STAR LC-10, NEU, 9 Nadeln, 144z Draft/36z NLQ	695,-
SOFTWARE	
fibuMAN Demodisk m. Handbuch, für ST oder PC	65,-
fibuMAN Einnahmen-Übersch. Rechn. ST 398,- / PC	898,-
fibuMAN Finanzbuchhaltung ... ST 768,- / PC	1468,-
fibu MAN Mandantenfähige FIBU . ST 968,- / PC	1968,-
Ramdisk C-N, Resetfestes ACC auch f. Harddisk	48,-

RUF (02 21) 21 17 61 Mo-Fr 10-12 u. 14-18 Uhr
5000 KÖLN 1 **HÄNDELSTRASSE 2—4**

Scanner ST

Scanner I zum Prg STAD 1.0 mit Eingriff in den Rechner	159,-
Scanner I als Bausatz	99,-
Scanner II zum Prg STAD 1.2 ohne Eingriff im Rechner	198,-
Scanner III 2–16 Graustufen mit Software	278,-
Flachbett Scanner	2998,-
HAWK CP 14 mit Software	
Handy Scanner	798,-

Hardware ST

Drews BTX-Manager V.2.0 mit Interface	428,-
Logic Analyser 16 Kanäle 600 KHz 5V TTL	448,-
Rom-Port umschaltbar	
Steckplatzerweiterung	79,-
NEC P2200 24 Nadel Printer	858,-

Tel. 0 26 31 / 7 24 03
Btx *0 26 31 / 7 24 03#

Roland Vodisek Elektronik
Kirchstr. 13
5458 Leutesdorf

COMPY SHOP

Wir haben die neueste Software für Ihren ATARI ST!

Zum Beispiel:	DM
WIZWAL (Dt.)	69,90
DIZZY WIZZARD (Dt.)	69,90
FOOTBALL MANAGER	49,90
SPACE BALLER (Dt.)	29,90
SPACE QUEST 2	69,90
CHECKMATE (Dt.)	34,90
XNON	34,90
EYE (Dt.)	44,90
KING QUEST alle drei Teile zusammen	79,90

Und vieles mehr! Fordern Sie unsere kostenlose ST-Software-Liste an!

COMPY-SHOP
Gneisenaustraße 29 · 4330 Mülheim Ruhr
Telefon (02 08) 49 71 69

Public-Domain Softwarepakete

★ **5 Markendisketten MF 1 DD, doppelseitig formatiert und gefüllt mit guter Public-Domain Software**

Achtung Preisänderung!



Paketpreis nur noch DM 39,–
inklusive Porto und Verpackung!

1 PD01 – PD10 aus ST-Computer	2 PD11 – PD20 aus ST-Computer	3 PD21 – PD30 aus ST-Computer
4 10 PD-Disk aus Kalifornien	5 10 PD-Disk aus Kalifornien	6 10 PD-Disk aus aller Welt
7 10 PD-Disk aus Kalifornien	8 10 PD-Disk aus Kalifornien	9 10 PD-Disk aus Kalifornien
10 PD 31-PD 40 aus ST-Computer	11 PD41-PD51 aus ST-Computer	12 10 PD-Disk aus aller Welt
13 10 PD-Disk aus aller Welt	14 10 PD-Disk aus aller Welt	15 10 PD-Disk aus Kalifornien
16 PD 52 – 61 aus ST-Computer	17 PD 62 – 71 aus ST-Computer	18 10 PD-Disk aus Kalifornien

19 Paket Enthält PD 72 – 81 aus ST-Computer. Z. B. Videoarchiv ★ Maxidisk ★ Datebook ★ Periode ★ AESLIB ★ U_Boot ★ Fonts für PD 40 ★ Malprogramm ★ etc...

20 Paket Enthält PD 82 – 91 + Updates „U1" und „U2" aus ST-Computer, z. B. Skat, Grusel, Masterpainter, Starofix, Hypervoc, etc...

21 Paket Enthält PD 92 – 101 aus ST-Computer. Z. B. ST-Speech ★ ST-Klick ★ MACPAN ★ Andurlin ★ Minigolf ★ Trash ★ Sharp ★ Minenfeld ★ Missile und vieles mehr

22 Paket Enthält PD 102 – 111 aus ST-Computer. Z. B. Lohnsteuer ★ Aktie ★ Depot ★ Poster ★ Napoleon ★ Druckerutilities ★ Wordplus Utilities ★ Soundcomputer...

23 Paket Enthält 12 Disks aus Kalifornien. Protect schützt vor Viren ★ NeoCalender erstelle Kalender aus Ihren Bildern ★ neue Borders für Printmaster ★ ArtDeco Fonts für Degas Elite ★ Bannermaker ★ Turtel Harddiskbackup ★ Spiele etc...

24 Paket enthält PD-Service Disk 112-121 + Updates Z. B. World ★ Zeitmanager ★ Metropol ★ Dallas ★ Sherlock ★ Scribble ★ Uniterm 2.0a ★ Lattice ★ Tape Label ★ Wissen Sie es ★ Go_Up ★ Oelimperium ★ uvm...

25 Paket Enthält PD-Service Disk Nr. 122 – 131 Z. B. PD_Bolo ★ CAD 2 ★ Horrorschloß ★ Hyperformat ★ Cube Hack ★ Inversi ★ Planet ★ Chemlib + einige Updates

Gewünschte Pakete ankreuzen u. Bestellung einsenden an:

IDL-Software
– Public Domain –
Alkmaarstraße 3 · 6100 Darmstadt 13

- ☐ Info-Diskette Nr. 5 DM 5,– PD-Liste + neue CLIP-ART + Belagerung, das Spiel um die Burg
- ☐ Scheck über DM ______ liegt bei. **Ich erhalte die Pakete verpackungs- u. versandkostenfrei**
- ☐ NACHNAHME (Nur Inland, zuzügl. DM 5,– NN-Gebühr)

Ganz Eilige bestellen einfach telefonisch: (0 61 51) 5 89 12

ST-COMPUTER PUBLIC DOMAIN SERVICE

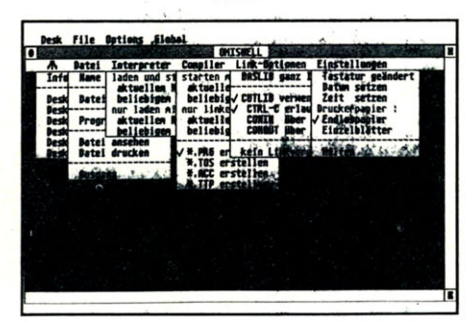
Liebe Leser,

es gibt viele Leser, die zuerst in die Public-Domain-Seiten schauen, wenn Sie die neue ST-Computer aufschlagen. Dies zeigt, wie wichtig unseren Lesern diese Software ist. Wir bemühen uns deshalb sehr, Ihnen aus den Einsendungen die besten herauszusuchen. Auch für unsere Mitarbeiter, die die Disketten kopieren, gilt Sorgfalt als erstes Gebot.
Leider kommt es oft vor, daß unsere Disketten von einigen Händlern unsachgemäß (z.B. ohne Verify oder auf schlechte Disketten) weiterkopieren oder sogar verändert werden.
Zu oft müssen wir hören, daß nicht bei uns bezogene Disketten Mängel aufweisen oder gar nicht laufen. Daß wir dafür nicht geradestehen können, dürfte verständlich sein. Wir können nur garantieren, daß direkt von uns verschickte Disketten fehlerfrei sind.
Da können Sie sicher sein.

Ihre ST-Computer Redaktion

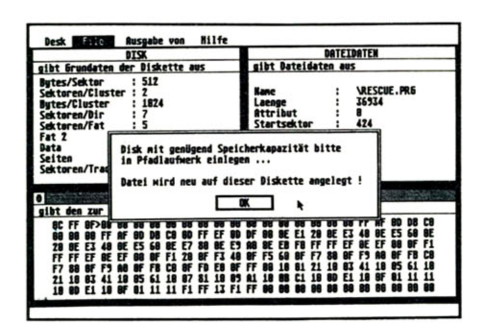
Neu in diesem Monat

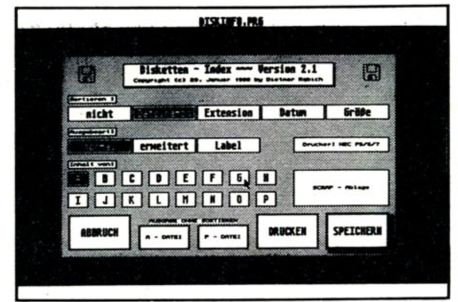




Utilities
- **GFA-Shell:** eine Shell für alle GFA-Basic Programmierer. Ruft Interpreter und Compiler aus der Menüleiste. Ein CLI und ein Sprite-Editor sind eingebaut. (s/w)
- **OMIKRON-Shell**: eine Shell für das OMIKRON.-Basic System. Aufruf von Interpreter, Compiler, Linker. Generiert PRG, TOS, TTP oder ACC. (s/w)
- **Disksave**: Speichert mit beliebigem Namen den kompletten Bildschirm per 'Alt+Help' anstatt zu drucken auf Diskette.
- **Apple-ST**: Datenübertragung Apple II—> ATARI ST. Enthält Programme für Apple- und GFA-Basic.





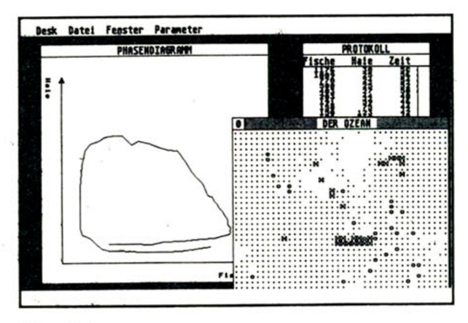


Utilities
- **HFORMAT**: Formatierprogramm als Accessory. Arbeitet im Hintergrund. d.h., daß z.B. Wordplus beim Formatieren weiterarbeitet.
- **Rescue:** Rettet gelöschte Files. Kein Grund mehr zum verzweifeln. (s/w)
- **Diskinfo:** Gibt das Directory sortiert auf Drucker (Liste oder Label) bzw. Datei aus. (s/w)
- **AccLoader:** Ermöglich zwischen mehr als 6 Accessories beim Systemstart auszuwählen. (s/w)



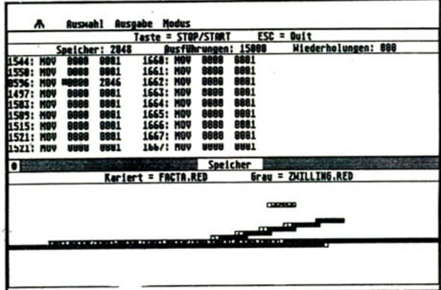
VIP-Macros
(professionelle Beispiele zum VIP-Kurs im Heft) (1MByte Ram)
- **Verein**: vollstaendige Vereinsverwaltung mit bis zu 212 Mitgliederstammsätzen. Serienbriefe, Lastschrifteinzug, Ueberweisungen Etikettendruck, Listendruck nach Kriterienauswahl.
- **Meister**: komplette Verwaltung einer Vereinsmeisterschaft.





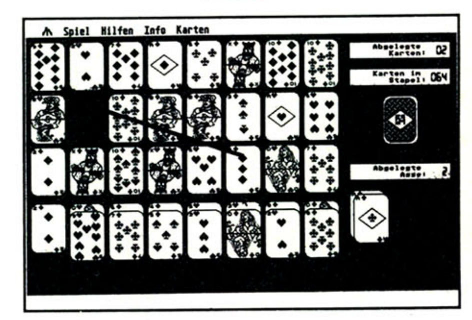
Simulationen
- **Wator:** Simulation eines biologischen Systems, am Zusammenleben von Haien und Fischen. Grafische Darstellung, Auswertung.
- **Life**: Ein sehr flexible Life-Version. Angabe der Fortpflanzungsregeln. (s/w)
- **Quicklife**: Rasend schnelles Life. Viele komplette Bildschirmberechnungen pro Sekunde. Für Leute in Zeitdruck. (s/w)





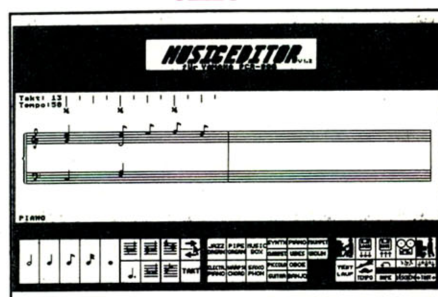
- **Codewars**: Programme kämpfen im Rechner gegeneinander (REDCODE). Multipass-Redcode-Assembler, Disassembler, grafische Anzeige. Max. 26 Programme gleichzeitig. Single-Step. Incl. ausführlich Einführung in die Theorie.
- **Kerne** 2.1: Erweiterte Version der 'Krieg der Kerne'. U.a. Grafische Darstellung des momentanen Spielfeldes. Erweiterte Befehle.





Spielesammlung
- **Galerie**: Wunderschöne Patience Version. Gefährlich für Redakteure, da sie damit von der Arbeit abgehalten werden.
- **Kreuz As**: Umsetzung des Spielautomat. Mut zum Risiko ist angesagt. (s/w)





- **Midi-Musik**: Komfortables Midi-Programm zur Kopplung eines Keyboards an den ST. Eingebauter Notendruck sorgt für die Ausgabe des Gespielten. (s/w)

ST-COMPUTER PUBLIC DOMAIN KLASSIKER

Der Redaktionsservice für unsere Leser.

Die nicht aufgeführten Disketten sind natürlich auch weiterhin erhältlich, nur wurde uns der Platz zu klein. Schauen sie dazu bitte in einer der vorherigen Ausgaben nach oder fordern Sie die Liste an (frank. Rückumschlag).

Diskette 7
- **XLISP 1.71:** (die Sprache der künstliche Intelligenz) incl. 54 KByte Dokumentation, Anleitung und Beispielen.

Diskette 11
- **PROLOG 10:** Die Sprache der 5. Genertion. Unterstützt GEM. incl. 95 KByte Anleitung und Beispiele.

Diskette 23
- **Das große deutsche Ballerspiel:** besonderes Aktionspiel (s/w)

Diskette 34
- **Assemblerpaket:** nach M 68000-Standard (2-Pass Assembler, Disassembler, Debugger, Editor)

Diskette 40
- **Deluxe Fontmaster ST:** Druckprogramm für Karten oder schönen Tabellen. Viele Zeichensätze, Proportionalschrift und Blocksatz, eigener Zeichensatzeditor.

Diskette 75
- Randvoll mit Zeichensätzen für Fontmaster (Disk 40).

Diskette 49, 50, 51
- **Volksforth 83:** Die drei Disketten beinhalten ein komplettes Forth-System für den ST. Alle Diskette werden benötigt.

Diskette 52
- **Molekül:** Darstellung chemischer Moleküle. In s/w und Farbe wobei in Farbe mit 3D-Effekt. Atom, Ionen oder Van der Waalsradius. Ein Molekül kann gedreht und sogar animiert werden.

Diskette 61
- **Hacomini** : Erzeugt Hardcopies in Miniaturformat (48mm * 68mm) (s/w).
- **Helper** : Diskmonitor und Druckerutility.
- **Bidipo** : Scrollt bis über 100 Bilder (4MB) über den Schirm. Für Diashows.
- **Datobert** : Grafische Darstellung. Linien- und Kuchengrafik, sowie in dreidimensionaler Balken- und Flächengrafik. (s/w)
- **Direct all:** Directorydruck.
- **Überweisung:** Überweisungsdruckprogramm.

Diskette 70
- **Orbit:** Satellitenbahnberechnung. Animation über der Weltkarte (s/w)
- **High End:** Programm zum Berechnen von Lautsprecherboxen aller Art. Lautsprecher und Boxenmaße. Datei für Speaker enthalten. (s/w)

Diskette 77
Utilitydisk
- **FCOPY 2.0:** Schnelles Kopierprogramm. Formatieren, Verify, Multiple Copy, Scan und Teilkopie.
- **Speeder:** Verdoppelt Schreib- und Lesegeschwindigkeit.
- **Filecopy:** Kopieren von Einzeldateien. Auch mit einer Floppy.
- **Formater:** Erhöht die Diskkapazität
- **Fastformat:** Formatiert Schnelladedisks.

Diskette 81
Malprogramm
- **Public Painter:** Superzeichenprogramm. Gepacktes Format, SNAPSHOT-Bilder, Lasso, Blockmanipulation, Vergrößern, Verkleinern, Drehen, Biegen, Verzerren, Überschneiden. Drucken von Bildausschnitten.(s/w)

Diskette 86
- **CARPET** : Berechnungsprogramm von 3D-Funktionen. Hidden-Line Algorithmus und Beleuchtungseffekten. Funktionseingabe und frei einstellbare Parameter. (s/w)
- **3D-PLOT** : Ähnliches Programm in compiliertem GFA-Basic, mit einigen Extra-Funktionen. (s/w)
- **FUNKTIONSPLOT** : Funktionseingabe im Programm, verschiebbare Achsen, mehrere Darstellungsarten (s/w)

Diskette 88
Druckertreiber
- **NEC-EMU** : Hardcopytreiber für NEC P5/P6/P7. Erzeugt Hardcopies in 12 verschiedenen Größen und versch. Kontrasten. Enthält Druckerspooler.
- **LQ 800** : Hardcopytreiber für EPSON 24 Nadeldrucker in drei verschiedene Größen.
- **NEC_CODE** : Ein Programm zur Einstellung des NEC P6 /P7 u.ä..
- **SIMPLE** : Komfortables Hardcopy-Programm mit vielen Extras.
- **STX-80** : Hardcopytreiber für diesen preiswerten Thermodrucker.
- **WORDPLUSTREIBER** : Druckertreibersammlung für folgende Drucker: NL10, FX1000, SP1000L

Diskette 90
Utilities
- **TOPS** : The Other Pascal Shell (für PascalPlus), TEMPUSfähig, erzeugt Crossreferenz. Kopieren, Löschen und Umbenennen von Files.
- **SIGNUMSHELL** : Eine Shell zu Signum. Umschaltung zwischen Signum Editor, den verschiedenen Druckprogrammen, den Zeicheneditoren per Menüleiste. Automatisches Install.
- **TEMPELMON 1.8** : DER weitverbreitete Monitor.
- **MAXIDISK 4** : Die komprimierende Maxidisk mit max. 4MByte Kapazität. Läuft auf ALLEN TOS-Versionen. Komprimiert automatisch ihren Inhalt. RESETFEST.
- **RAMDISK** : Gleiches Programm wie MAXIDISK, nur ohne Komprimieralgorithmus.
- **COPY** : Beide Ramdisk enthalten ein Autocopyprogramm zum kopieren bestimmter Files in Ramdisk.
- **RCS.WANDEL** : Wandelt *.h Files nach Basic und Assembler.

Diskette 93
- **ADR_2** : Semiprofessionelles Adressverwaltungsprogramm mit Serienbrieferstellung (mit und ohne 1st Word). Editierbare Druckeranpassung (Steuerzeichen, Zeichenwandlung). erzeugt Adress- Telefon und Geburtstagslisten. Druckt Labels, Etiketten nach freidefinierbarem. Format. (s/w)
- **DAT_TEXT** : Adressverwaltung für Mehrpersonenzugriff auf eine Datei. Datenzugriff nur durch Passwort (s/w)

Diskette 94
Spieledisk
- **MACPAN** : Eine schnelle Pacman-Version für den ST. Per Joystick ins Labyrinth auf der Suche nach der Powerpills. Mit Empfehlung der Redaktion.
- **SOLITÄR** : Dreidimensionales Solitär mit sehr ansprechender Grafik.
- **GALAXY + SPRENGMEISTER** Brettspiele zur Förderung der Gehirnaktivität (s/w)

Diskette 100
- **ST-SPEECH** : Sprachausgabe auf Softwarebasis. Wandelt englischsprachige Texte automatisch in verständliche Lautschrift. (s/w)
- **FORTRAN SHELL** : Eine Shell für Fortran 77. Bequemes Aufrufen von Compiler, Linker, und anderen Programmen.
- **LIFE IS LIFE** : Life-Simultionsprogramm. Sechs verschiedene Feldgrößen, Drehen, Verschieben, Kopieren von Blöcken wie im Malprogramm, Festlegen der Fortpflanzungsregeln, hohe Geschwindigkeit. (s/w)
- **FASTLIFE** : Life-Simulation in Volksforth. Wahnsinnige Geschwindigkeit. (s/w)
- **GEMFRAC** : Berechnung fraktaler Landschaften in Farbe und s/w.
- **SHARP** : Basic Programme für Sharp-Basicrechner. Geeignet für Direktüberspielung mit Interface.

Diskette 104
- **ST_CALC** :Kalkulationsprogramm mit vielen Features. 2600 Zellen. Variabler Bildaufbau. Formeleingabe. FolgendeOperationen: (), *, +, -, ^, /, SUM, AVE, STA, MIN, MAX, ABS, INT, RND, LOG, EXP, CLOG, SQR
- **LOHNSTEUER** : Berechnet die Lohnsteuer für die Jahre 1977 und 1988
- **ANLAGE** : Verwaltungsprogramm von Anleihen, Aktien, Investmentanteile, u.a. (s/w)
- **DEPOT** : Aktiendepot. Nützlichesprogramm für Finanzgenies.
- **AKTIE** : Auch dieses Programm dient der Aktienverwaltung.

Diskette 108
Druckerutilities (P6 u.a.)
- **SIG PIC** : Grafikeinbindung in das alte SIGNUM! V1.0. Versch. Bild-Größen.
- **ZS Editor 24** : Zeichensatzeditor für 24-Nadeldrucker. Down-Load. Für alle Programme (z.B. Wordplus). (s/w)
- **MANAGER** : Accesoir zum Einstellen eines P5/6/7-Druckers.
- **PRINT_OUT** : Ausdruck von Source-Code (Pascal oder C) mit Zeichenanpassung und beliebige Schriftartwechsel.
- **POSTER** : Druckt Bild in vierfacher Größe. (s/w)

Diskette 109
Wordplus-Utilities u.a.
- **INHALT** : Erzeugt Index-Liste von 1stWord-Texten.
- **INDEX_PLUS** : Erzeugt Liste von Seitenindex, Zeilenlinealen und allen Bildern.
- **BT_Konvert** : Konvertiert 1st_Word-Texte nach Beckertext.
- **TYPEWRITER** : Schreibmaschinenkurs in über 40 Lektionen.(s/w)
- **MORSE** : Morsetrainingsprogramm.

Diskette 110
Utilities
- **R_COPY** : Automatischen Backup von Ramdisk auf Diskette. Incl. Source in Pascal.
- **PASHELP** : Crossreferenzprogramm für Pascal.
- **UNIDRUCK** : Druckprogramm mit Sonderzeichenanp., mehrere Drucker. Zeilenummerierung. Incl. Source in Pascal.
- **FONT** : Programm zum Einbinden von DEGAS-Fonts in Pascal-Programme. Incl. Source in Pascal.

Diskette 111
Utilities
- **BASICREF** : Crossreferenz für GFA-Basic. Incl. Source in GFA-Basic. (s/w)
- **MSK_Edit:** Maskeneditor für GFA-Basic. Generiert Quelltext (Form Input). Incl. Source in GFA-Basic. (s/w)
- RETTEN: Utilitie für lange Dateien. Incl. Source in Fortran.
- SECO:Bildausschnittsbibliotheksverwaltung (für PUT GET von GFA). Incl. Source in GFA-Basic.

Diskette 112
Erdkunde
- **WORLD:** Lernprogramm. Fragt alle Länder und Hauptstätte der ganzen Weltab. Landkarten von BRD, USA, Mittelamerika, Südamerika, Europa, Asien, Afrika und Ozeanien. (s/w)

Diskette 113
- **ZEITMANAGER** : Semiprofessionelle Terminplanverwaltung. Eigener Desktop, Terminerinnerung, Listendruck, Timingverwaltung, fixe Termine, Kalender. Schaltjahrfest. (s/w)

Diskette 114
Spiele
- **METROPOL** : Wirtschaftsspiel. RettenSie die Wirtschaft ihres Landes. 'Auf der Suche nach der Wende'.(s/w)
- DALLAS : Öl, Öl, Öl, Geld, Geld, Geld, JR, JR, JR (f)

Diskette 115
Spiele
- **SHERLOCK:** Klären Sie als Holmes den Mord.
- **EL BOZO CITY:** Textadventure mit vielen Gags.
- **SCRIBBLE** : Schöne Scrabble-Version für den ST. (s/w)
- **KREUZWORT** : Generierung eines Kreuzworträtsels. (s/w)

Diskette 116
- **µEMACS 3.8** : Deutsche Version. Eigene Macrosprache, Textverschlüsselung, Mailmerge, mehrere Screens, u.v.a.m.

Diskette 117
Terminalprogramme
- **UNITERM** : Exclusives Terminalprogramm. VT200, VT102, VT100, VT52,4010, u.a. Softscroll, Grafikübertragung, Funktionstastenbelegung,..

Diskette 118
Utilities
- **M_COPY** : Universelles Kopierprogramm bzw. Diskmonitor.- LOADER : Aktiviert oder deaktiviert Accesoiries bzw. Autostartprogramme.
- **VIEW** : Programm zum Anzeigen von ASCII-Files. Vor- und Rückblättern per Tastendruck.
- **SUPERSHELL** : Universelle Shell. Aufruf beliebiger Programme aus der Menüleiste. Source in GFA-Basic.
- **LATTICE** : diverse Utilities: HEXDUMP, Ausdruck mit Zeilennummern, Symboltabellen, Archivierung und Pflege von Modulbibliotheken.

Diskette 119
Etiketten Drucker
- **ST LABEL:** Etikettendrucker.
- **LABEL** : Komfortabler Disketikettendrucker. Übernahme eines Bildes per Mausklick. Eigene Bilder integrierbar. (s/w)
- **PASTE ADRESS** : Adressaufkleber in Miniformat (1.5cm*2.5cm)

Disketten 120
- WISSEN SIE ES? : Quizspiel mit vielen Fragen aus verschiedenen Gebieten. Ähnlich 'Trivial Pursuit'. Viele Fragen aus verschiedenen wissensgebieten, eigene integrierbar. (s/w)

Diskette 121
Spiele
- **GO_UP** : Loderunner-ähnliches Spiel. Auf vielen Mauern, Leitern und Seilen müssen Sie den Verfolgern entkommen. Viele Level. Editor zum Entwickeln eigener Levels enthalten. (s/w)
- **MAZE_EDIT** : Editor zu dem Spiel 'MIDI MAZE'.
- F1 MANAGER : Verwaltung eines Formel-Eins-Rennstall. (s/w)
- **OELIMPERIUM** : Managen einer Ölfirma. (s/w)

Diskette 122
Spiel
- **PD_BOLO:** Sie kennen Arkanoid, dann sollten Sie auch PD_Bolo kennen. Interessante Break-Out Variante. (mind. 1 MByte)

Diskette 123
Utilities
- **Hyperformat** Formatierprogramm von Claus Brod.
- **FSELECT** Die neue Fileselectorbox. Wird fest installiert. Kommt bei jedem Aufruf. Warum nicht gleich so ATARI?
- **Diskmon** Diskettenmonitor. (s/w)
- **PC_Ass** Assembler für Sharp-Taschencomputer.

Diskette 124
Chemie
-**CHEMLIB** : Dateiverwaltung für chemische Elemente (s/w)

Diskette 125
Zeichenprogramm
- **CAD 2:** Objektorientiertes Grafikprogramm. z.B. mit Zoom ohne Auflösungsverlust.(s/w)

Diskette 126
- **CUBE HACK:** Terminalprogramm mit integrierter, umfangreicher Programmiersprache. Anwendungen: z.B. Automatisches Einloggen in Mailboxen.
-**OR Spread:** Spreadsheetähnliches Programm zur Linearoptimierung, Netzplanentwicklung, Kürzeste Wege Berechnung. (s/w)

Diskette 127
Wissenschaftliche Anwendungen
- **Planet:** Himmelskörperberechnungen. (Grafische Darstellung des Sternenhimmels, Horizont). Animation.
- **Turing:** Realisierung eines Turing-Maschinen-Modells (s/w)
- **Zust:** Analyse und Simulation linearer Regelkreise (s/w)

Diskette 128
Spielesammlung
- **Horrorschloß:** Der Weg durch viele Gänge verbirgt viele Gefahren.
- **Inversi:** Das bekannte Spiel, gegen den Rechner oder zu zweit (s/w)
- **Invation:** Space Invadors mit 3-D Vektorgrafik. (s/w)
- **Zarge:** Ein 32000K-Bild wird gepuzzelt (s/w)

(s/w) - Nur Monochrom
(f) - Nur Farbe
kein Kürzel - Farbe und Monochrom

Sonderdisks

Die folgenden Programme sind nicht Public-Domain.
Sie können aber bei uns bezogen werden.

A.) TOS : Die letzte Disketten-Version vom 6.2.1986. Ältere Versionen laufen nicht problemlos.
Unkostenbeitrag samt Diskette DM 15.-
B.) RCS : Das Resource-Construction Set aus dem ATARI-Entwicklungspakets. Unverzichtbar bei der GEM-Programmierung.
Unkostenbeitrag samt Diskette DM 15.-
Bis auf die Höhe des Unkostenbeitrages gelten die gleichen Versandbedingungen wie bei der Public-Domain Software.

VERSANDBEDINGUNGEN

Sämtliche Disketten können ab dem Erstverkaufstag der ST-Computer direkt bei der Redaktion bezogen werden. Wir haben für Sie den schnellstmöglichen Versandservice eingerichtet. Lieferung innerhalb einer Woche.

1. Schriftliche Bestellung
- Der Unkostenbeitrag für eine Diskette beträgt DM 10.-
- Bezahlung nur per Scheck oder Nachnahme (Im Ausland nur Vorrauskasse möglich)
- zuzüglich DM 5,00 Versandkosten (Ausland DM 10.-)
- bei Nachnahme zuzüglich DM 3,70 Nachnahmegebühr
- Legen Sie bitte, falls zur Hand, einen Aufkleber mit Ihrer Adresse bei.
- Bitte fügen Sie keine anderen Bestellungen oder Anfragen bei

Die Bezugsadresse lautet:
'MERLIN'-Computer GmbH
ST-Computer Redaktion
'PD-Service'
Postfach 5969
D-6236 Eschborn

2. Anruf genügt
'MERLIN'-Computer GmbH
ST-Computer Redaktion
Tel.: 0 61 96 / 48 18 11
Von Mo-Fr 9.00 bis 17.00 Uhr

Die Bezugsadresse lautet:
'MERLIN'-Computer GmbH
ST-Computer Redaktion
'PD-Service'
Postfach 5969
D-6236 Eschborn

Bei Fragen bezüglich der Programme stehen wir Ihnen gerne zur Verfügung.

Immer up to date

Mit dieser Sparte wollen wir allen unseren Lesern die Möglichkeit geben, sich über die neuesten Programm-Versionen zu informieren. Angegeben werden die aktuelle Versionsnummer, ein eventueller Kopierschutz, die Bildschirmauflösungen und der Speicherbedarf. Softwarefirmen ist es somit möglich, die ST-Computer-Leser über ihre Updates zu informieren.

Programmname	Version	Daten	Programmname	Version	Daten
AnsiTerm	1.1	N	1st_Lektor	1.2	H
Binlook	1.0	N HML	Lisp Complete	1.01	N HM
BTX für ST	1.0	N H 1M	Lock_it	1.0	J ML
Crypt_it	1.0	J HML	Megamax C	1.1	N HML
Flash-Cache/Flash-Bak	1.0	N HM	Mega Paint	1.0	N H 1M
Flexdisk	1.1	N HM	Megamax Modula 2	1.0	N HM
1st_Freezer	1.0	N HML 1M	Micro C-Shell	2.70	N HM
GFA-Artist	1.0	N L	MT C-Shell	1.10	N HM 1M
GFA-BASIC 68881	1.3	N HML	Multi ST	1.0	N HML 1M
GFA-Compiler	2.02	N HML	Musix32	1.01	J H
GFA-Draft	2.1	N	Omikron BASIC Compiler	2.0	N HML
GFA-Draft plus	1.5	N	Omikron BASIC Interpreter	2.0b	N HML
GFA-Interpreter	2.02	N HML	PC ditto Euroversion	3.64	N HML
GFA-Objekt	1.1	N HM	Pro Sound Designer	1.2	L
GFA-Starter	1.1	N HML	Pro Sprite Designer	1.0	L
GFA-Vektor	1.0	N	Search!	1.31	N HM
Hard Disk Accelerator	1.0	N HML	Signum! zwei	2.0	N H
Hard Disk Toolkit	1.04	N HM	1st_Speeder	1.01	N HML
Harddisk Utility	1.04	N HM	1st_Speeder 2	1.0	N HML 1M
Imagic	1.0	N HML	STAD	1.2	N H
Intelligent Spooler	1.01	N HML	ST Pascal plus	2.02	N HM
K-Resource	1.1	N HM	Tempus	2.0	N HM
Laser Deluxe	1.0	N HML 1M	VSH Manager	1.11	N HML 1M

Irrtum vorbehalten
Daten-Legende : N = kein Kopierschutz, J = Kopierschutz, H = hohe Auflösung, M = mittlere Auflösung, L = niedrige Auflösung, 1M = mindestens 1 Megabyte Speicher notwendig

INSERENTEN-VERZEICHNIS

In der nächsten ST-Computer lesen Sie unter anderem

Was gibt's Neues von ATARI?

Alle warten gespannt auf die CeBIT '88 in Hannover, um zu erfahren, was an neuen Produkten aus dem Hause ATARI zu erwarten ist. Zu sehen sein, wird sicherlich der Transputer-Rechner sein und wahrscheinlich ist das CD-ROM da. Doch was ist mit einem neuen 32-Bit-Rechner mit einem 68020- oder 68030-Prozessor? ATARI Deutschland Chef Alwin Stumpf meinte nur: "Wir sind immer für Überraschungen gut !" Was nun wirklich zu sehen war, berichten wir in der nächsten Ausgabe.

... doch nicht nur ATARI ist auf der CeBIT!

Natürlich konzentrieren sich die meisten ST-Besitzer auf ATARI, doch was wäre ein Rechner ohne Software, Drucker und andere Peripheriegeräte. Allein auf dem ATARI-Stand werden an 50 Tischen Produkte aus deutschen und anderen Landen frisch auf den Tisch angeboten. Wir berichten ausführlich von der CeBIT '88, was am ATARI-Stand und sonst noch auf dieser wichtigen Messe zu finden war.

Laservisionen

Wer bis jetzt mit dem ATARI Laserdrucker SLM 804 gearbeitet hat, mußte leider sehr schnell gewisse Mängel an der Treibersoftware feststellen. Um dieses Manko zu beheben, veröffentlichen wir in zwei Teilen ab der nächsten Ausgabe einen neuen Laserdruckertreiber, mit dem sowohl Grafik- als auch Text problemlos auszudrucken sind. Alles über die Programmierung des Lasers und einen Grafiktreiber kann man in der Mai-Ausgabe finden.

Megamax Modula-2

Lange schon ist es angekündigt, doch jetzt ist es endlich fertig. Nach Megamax C kommt jetzt mit Megamax Modula-2 eine zweite Sprache aus der Softwareschmiede Application Systems /// Heidelberg. Was bringt dieses Modula-2 gegenüber den bereits auf dem Markt erhältlichen Konkurrenzprodukten.

Video und ST

Wer es bis jetzt vergeblich probiert hat, seinen ATARI ST zum Erstellen von Videovorspännen, Trickfilmen etc. zu benutzen, dem kann jetzt geholfen werden. Unser Hardwareprojekt FBAS-Wandler ermöglicht es, das FARBBILD des STs auf jedem Videorecorder aufzunehmen bzw. auf Fernsehern mit Videoeingang darzustellen.

Änderungen vorbehalten !

Die ST-Computer Ausgabe 5/88 erscheint am 29.4.1988

Fragen an die Redaktion

Ein Magazin wie die ST-Computer zu erstellen kostet sehr viel Zeit und Mühe. Da wir ja weiterhin vorhaben, die Qualität zu steigern (ja, auch wenn das manchmal danebengeht), haben wir Redakteure ein großes Anliegen an Sie, liebe Leserinnen und Leser:

Bitte haben Sie Verständnis dafür, daß Fragen an die Redaktion in Zukunft nur noch zu bestimmten Zeiten beantwortet werden können. Wir stehen Ihnen zu folgenden Terminen telefonisch zur Verfügung:

Donnerstag von 14 - 17 Uhr

Vielen Dank für Ihr Verständnis

Ihre Redaktion

Impressum

ST Computer

Chefredakteur:
Uwe Bärtels (UB)

Redaktion:
Uwe Bärtels (UB)
Harald Egel (HE)
Marcelo Merino (MM)
Harald Schneider (HS)

Redaktionelle Mitarbeiter:
Claus Brod (CB) Claus P. Lippert (CPL)
Stefan Höhn (SH) Markus Nerding (MN)
Raymund Hofmann (RH) Chr. Schormann (CS)
Oliver Joppich (OJO) Andreas Suchy (AS)
H.P.Labude (HP) Jörg Wilhelm (JW)
Jürgen Leonhard (JL)

Autoren dieser Ausgabe:
R.Bager L.Preßler
D.Brockhaus D.Rabich
I.Brümmer A.Esser
Chr.Schmitz-Moormann
P.Fischer M.Schuhmacher (MS)
Dr.P.Härtel H.D.Schultz
Dr.V.Kurz M.Siebler
F.Matthes R.Tolksdorf

Public Relations:
Claus P. Lippert (Leitung)
D.dela Fuente (UK)
L.Hennelly (Nordamerika)

Redaktion: "Merlin" Computer GmbH
Postfach 59 69
Industriestr. 26
6236 Eschborn
Tel.: 0 61 96/48 18 11
FAX: 0 61 96/4 11 37

Verlag:
Heim Fachverlag
Heidelberger Landstr. 194
6100 Darmstadt 13
Tel.: 0 61 51/5 60 57
FAX: 0 61 51/5 56 89 + 5 60 59

Verlagsleitung:
H.J.Heim

Anzeigenverkaufsleitung:
U.Heim

Anzeigenverkauf:
K.Magaritis

Anzeigenpreise:
nach Preisliste Nr.3, gültig ab 1.1.88
ISSN 0932-0385

Grafische Gestaltung:
Fabian & Mayer

Fotografie:
R.Spirandelli, Archiv

Produktion:
K.H.Hoffmann, B.Failer, S.Failer

Druck:
Ferling Druck

Lektorat:
V.Pfeiffer

Bezugsmöglichkeiten:
ATARI-Fachhandel, Zeitschriftenhandel, Kauf- und Warenhäuser oder direkt beim Verlag

ST Computer erscheint 11 x im Jahr
Einzelpreis: DM 7,-, ÖS 56,-, SFr 7,-
Jahresabonnement: DM 70,- Europ. Ausland : DM 90,- Luftpost : DM 120,-

Manuskripteinsendungen:

Programmlistings, Bauanleitungen und Manuskripte werden von der Redaktion gerne angenommen. Sie müssen frei von Rechten Dritter sein. Mit seiner Einsendung gibt der Verfasser die Zustimmung zum Abdruck und der Vervielfältigung auf Datenträgern dem Heim Verlag. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte wird keine Haftung übernommen.

hardware
software
organisation
service

CSF GMBH

Heeper Str. 106 – 108, 4800 Bielefeld 1, 05 21 / 6 16 63

Kein Kabelsalat mehr mit dem Gehäuse für ATARI ST

- **Zentrale Stromversorgung für alle Geräte einschl. 2 Drucker**
- **Einbaumöglichkeit von 2 Diskettenlaufwerken**
- **Rechner (Tastatur) kann komplett unter das Gehäuse geschoben werden (Staubschutz)**
- **Massives Blechgehäuse**

ATARI ST-Gehäuse erhalten Sie bei den autorisierten Fachhändlern

NEU: DM 149,–

NEUES VON CSF

- **Speichererweiterungen auf 1 MB,**

steckbar oder zu Löten
für ATARI 520 ST, 260 ST, 520 STM

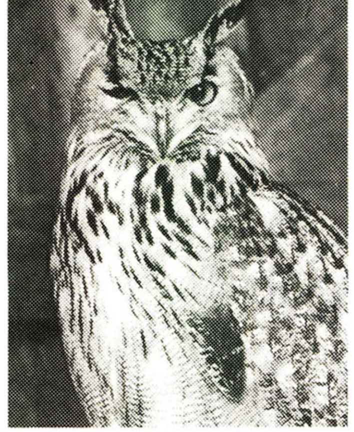
225,– DM

- **steckbar** (ohne jegliche **Lötarbeiten**)
- läuft auch auf dem 520 STM
- enorme Zeitersparnis durch einfache, bebilderte Einbauanleitung
- kein Flimmern nach der Erweiterung (durch separate, geglättete Spannung an der zweiten RAM-Bank)
- sensationeller Preis
- Bei Bestellungen bitte angeben: nur für 260 und 520 ST
  - ☐ Speichererweiterung steckbar
  - ☐ Speichererweiterung zum Löten

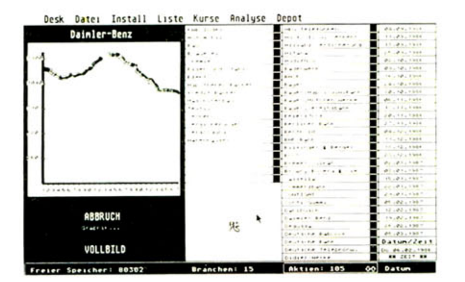
Zu beziehen:	In der Schweiz:	In Österreich:
Direkt bei CSF, Bielefeld **Tel. 05 21 / 6 16 63** **Bei allen ATARI-Händlern**	**SENN Computer AG** **Langstr. 31 · CH-8021 Zürich** **Tel. 01-241 73 73**	**Institut für Datenverarbeitung und Organ. Ges.mbH** **Rehberger Hauptstr. 95 · A-3503 Krems** **Tel. 0 27 32 - 7 05 81**

Alle Preise sind unverb. empf. Verkaufspreise

Börsenspekulanten fordern:



1. **Aktienanalyse** für den privaten und professionellen Aktienanleger auf der Basis der **Relativen Stärke und Volatilität** von Aktienkursen.
2. **Charts** wahlweise über Bildschirm und Drucker.
3. **Mischen von Charts** einer Aktie und Branche zum Vergleich.
4. **Kurseingabe nur einmal wöchentlich (!).** Als Basis dienen die letzten 27 Wochenschlußkurse.
5. Umfangreiche Bereinigungsfunktion bei Kapitalveränderungen von Aktien.
6. **Listen** der Relativen Stärke mit Mittelwert von Kursen und Analyse der Veränderung der Rangfolge der Relativen Stärke über 3 Wochen (wahlweise Bildschirm oder Drucker).
7. Zu jeder Aktie können **Höchst- und Tiefstkurse** sowie eigene **Notizen** eingegeben werden.



8. **Depotverwaltung** für den privaten und semiprofessionellen Aktienanleger über ein **bedienungsfreundliches Karteikartensystem** auf dem Bildschirm mit **Gewinn- und Verlustrechnung** Brutto.
9. **Depotübersicht** wahlweise auf Bildschirm oder Drucker aufgeschlüsselt nach Einzelgattung oder als Gesamtübersicht.
10. Umfangreiche **Lösch- und Änderungsfunktion** für die Daten und **weitere Extras,** z.B. Bildschirmverdunkeln bei Arbeitspausen.
11. Je nach Arbeitsspeicher können bis zu 20 Branchen und 500 Aktien erfaßt werden (Warnung bei Speicherproblemen).



Siemensstraße 16, 4630 Bochum 1
Telefonische Bestellungen: 023 25 / 6 08 97

Schweiz: Senn Computer AG
Langstr. 31, CH-8021 Zürich

Österreich: Computershop Rittner
Hauptstr. 34, A-7000 Eisenstadt

BENELUX: G DATA BeNeLux
Postbus 70, NL-2000 AB Haarlem



DM 398,—
unverbindliche Preisempfehlung